A STUDY OF NUMERICAL METHODS OF SOLUTION OF THE EQUATIONS OF MOTION OF A CONTROLLED SATELLITE UNDER THE INFLUENCE OF GRAVITY GRADIENT TORQUE

BY

JOE F. THOMPSON
JOHN C. MCWHORTER
SHAHID A. SIDDIQI
SAMUEL P. SHANKS
A STUDY OF NUMERICAL METHODS OF SOLUTION OF THE EQUATIONS OF MOTION
OF A CONTROLLED SATELLITE
UNDER THE INFLUENCE OF GRAVITY GRADIENT TORQUE

by

Joe F. Thompson
John C. McWhorter
Shahid A. Siddiqi
Samuel P. Shanks

Report Number EIRS-ASE-74-1

Prepared by

Mississippi State University
Engineering and Industrial Research Station
Department of Aerophysics and Aerospace Engineering
Mississippi State, Mississippi 39762

Under Contract NAS8-28833

for

NATIONAL AERONAUTICS and SPACE ADMINISTRATION
George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama 35812

July 1973
TABLE OF CONTENTS

ABSTRACT. ......................................................... iii
INTRODUCTION. ................................................. 1
CHAPTER I - EQUATIONS OF MOTIONS. ......................... 3
CHAPTER II - NUMERICAL EXPERIMENTATION PROCEDURE. .... 9
CHAPTER III - RESULTS OF COMPARISON ....................... 12
  Runge-Kutta Methods. .................................... 12
  Multi-step Methods .................................... 13
  Extrapolation Methods ................................ 15
CHAPTER IV - SURVEY OF TECHNIQUES FOR FURTHER
  CONSIDERATION. .............................................. 19
  Quadrature Methods ...................................... 19
  Multi-value Methods ..................................... 22
  Non-Polynomial Interpolates ............................. 26
  Iterative Methods ....................................... 27
  Transformation Methods ................................ 27
  Higher-Order Equations ................................ 27
  Conclusions and Directions .............................. 28
TABLE I .......................................................... 30
TABLE II .......................................................... 31
Figure 1. $W_3$ and $q_3$ up to $T = .25$ (quarter of an orbit). 32
Figure 2. Errors in $W_3$ and $q_3$ versus time, up to $T = .04$ . 33
Figure 3. RMS error in $W_3$ versus step size .................. 34
Figure 4. Comparison of ERPESR and BSRESH Extrapolation
  Methods - Time Step Effects. ............................ 35
Figure 5. Comparison of ERPESR and BSRESH Extrapolation
  Methods - Tolerance Effects. ............................ 36
BIBLIOGRAPHY. .................................................. 37
APPENDIX I ...................................................... I-1
APPENDIX II ..................................................... II-1

//
ABSTRACT

Numerical methods of integration of the equations of motion of a controlled satellite under the influence of gravity-gradient torque are considered. The results of computer experimentation using a number of Runge-Kutta, multi-step, and extrapolation methods for the numerical integration of this differential system are presented, and particularly efficient methods are noted. A large bibliography of numerical methods for initial value problems for ordinary differential equations is presented, and a compilation of Runge-Kutta and multi-step formulas is given. Less common numerical integration techniques from the literature are noted for further consideration.

This report was prepared by Department of Aerophysics and Aerospace Engineering, Mississippi State University under Contract NAS8-28833 for the George C. Marshall Space Flight Center of the National Aeronautics and Space Administration.
INTRODUCTION

The integration of the equations of motion describing the dynamics of a body in orbit, as affected by various perturbing forces and the corrective action of control systems, normally requires the use of hybrid computers because of the difficulty and time involved in integration of high frequency components by present numerical techniques.

It is desirable, however, to be able to describe the vehicular dynamics and the vehicle response to control action numerically in order to take advantage of the advanced development and ease of use of digital computers, particularly on board the vehicle. This can only be achieved by improved mathematical analysis of the numerical integration of the initial value problem with simultaneous ordinary differential equations. While the numerical techniques that have become standard are adequate for the integration of such systems when only low frequency modes are involved, the efficient analysis of high frequency modes requires the development of new approaches.

It is therefore necessary to develop further the mathematical analysis of the numerical integration of systems of simultaneous ordinary differential equations as an initial value problem with specific application to the equations describing the vehicular dynamics of a controlled body in orbit. The ultimate goal of the present project is to develop particularly efficient numerical integration schemes for the equations of motion of a flexible orbiting body rotating under the influence of gravity gradient and control torques.

In the present investigation an extensive bibliography of papers dealing with the numerical solution of systems of ordinary differential
equations has been compiled and is included in this report. Many different methods of solution have been obtained therefrom to be compared. In the present effort comparisons of a large number of Runge-Kutta, multi-step, hybrid, and extrapolation methods have been made using the equations of motion of a rigid satellite in circular orbit rotating under the influence of gravity gradient and control torques to a fixed attitude, and the results of these comparisons are reported herein. Further effort is required to extend the comparison to other types of methods, and to compare all methods in regard to the equations of motion of non-rigid satellites.
CHAPTER I

EQUATIONS OF MOTION

For a rigid space vehicle in orbit, six ordinary differential equations are required to specify its spatial motion—three equations for rotations and three for displacements. If the vehicle is considered non-rigid, more equations (partial differential equations) are required to specify the deformations of the body.

In this study the body is considered rigid. The equations of motion are formulated in terms of quaternions, these being functions of the direction cosines of the body axes. The introduction of these quaternions results in seven first order ordinary differential equations which specify the motion of the body.

The vehicle is in orbit and the motion is assumed to be affected only by earth's gravitational field. The space vehicle is to be kept attitude fixed, that is, its direction should be invariant for all time. Any deviations from this fixed direction are to be corrected by an on-board control system. Deviations from the fixed attitude occur because of the gravity gradient and imperfections in the launch process. These deviations are indicated by the quaternions.

The seven first-order ordinary differential equations, in matrix form are:

\[ \dot{\omega} + \omega \times \dot{\omega} = \frac{3GM}{|r|^5} r \times \dot{r} + T \]  
\[ \dot{\mathbf{q}} = -\frac{1}{2} \Omega(w)\mathbf{q} \]
where
\[ \mathbf{w} = \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} \]
is the angular velocity vector,
\[ \dot{\mathbf{w}} \]
is the time rate of change of \( \mathbf{w} \),
\[ \mathbf{q} = \begin{pmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{pmatrix} = \begin{pmatrix} e_1 \sin \psi/2 \\ e_2 \sin \psi/2 \\ e_3 \sin \psi/2 \\ \cos \psi/2 \end{pmatrix} \]
is the quaternion vector,
\[ \dot{\mathbf{q}} \]
is the time rate of change of \( \mathbf{q} \),
\( \psi \) is the angle of rotation of the body's rotation vector \( \mathbf{e} \) from a fixed attitude,
\( \mathbf{T} \) is the control torque vector, a function of \( \mathbf{w} \) and \( \mathbf{q} \),
\[ \mathbf{I} = \begin{pmatrix} i_{11} & i_{12} & i_{13} \\ i_{21} & i_{22} & i_{23} \\ i_{31} & i_{32} & i_{33} \end{pmatrix} \]
is the moment of inertia matrix,
\( \mathbf{r} \) is the radius vector of the orbit referred to the vehicle fixed axes and is related to the inertial axes, fixed to the center of the earth, by a transformation matrix \( \mathbf{D} \),
\( G \) is the universal gravitational constant,
\( M \) is the mass of the earth,
\( \mathcal{D}(\mathbf{w}) \) is an asymmetric matrix function of \( \mathbf{w} \) as defined below:
\[ \Omega(w) = \begin{pmatrix}
0 & -w_3 & w_2 & w_1 \\
w_3 & 0 & -w_1 & w_2 \\
-w_2 & w_1 & 0 & w_3 \\
-w_1 & -w_2 & -w_3 & 0
\end{pmatrix} \]

Since \( I \) is non-singular, we may write (1.1) as

\[ \dot{w} = I^{-1} \left[ \frac{3GM}{|r|^5} r \times I^*r - w \times I^*w + T_c \right] \tag{1.3} \]

Evaluating the matrix cross products and defining an asymmetric matrix function \( F(\alpha) \), where \( \alpha \) is any vector, by

\[ F(\alpha) = \begin{pmatrix}
0 & -\alpha_3 & \alpha_2 \\
\alpha_3 & 0 & -\alpha_1 \\
-\alpha_2 & \alpha_1 & 0
\end{pmatrix} \]

we have

\[ \dot{w} = I^{-1} \left[ \frac{3GM}{|r|^5} F(r)I^*r - F(w)I^*w + T_c \right] \tag{1.4} \]

\[ \dot{q} = -\frac{1}{2} \Omega(w) q \tag{1.5} \]

The transformation of \( r_I \) (inertial axes) to \( r \) (vehicle axes) is given by

\[ r_I = R_0 \begin{pmatrix}
\cos w_0 t \\
\sin w_0 t \\
0
\end{pmatrix} \tag{1.6} \]
\[ r = D r_I \]

where

\[
D_{11} = q_1^2 - q_2^2 - q_3^2 + q_4^2
\]

\[
D_{12} = 2(q_1 q_2 + q_3 q_4)
\]

\[
D_{13} = 2(q_1 q_3 - q_2 q_4)
\]

\[
D_{21} = 2(q_1 q_2 - q_3 q_4)
\]

\[
D_{22} = -q_1^2 + q_2^2 - q_3^2 + q_4^2
\]

\[
D_{23} = 2(q_2 q_3 + q_1 q_4)
\]

\[
D_{31} = 2(q_1 q_3 + q_2 q_4)
\]

\[
D_{32} = 2(q_2 q_3 - q_1 q_4)
\]

\[
D_{33} = -q_1^2 - q_2^2 + q_3^2 + q_4^2
\]

The control torque vector is given by

\[
\mathbf{T}_c = \mathbf{A} \mathbf{q} + \mathbf{b} \mathbf{w}
\]

where

\( \mathbf{A} \) is a control matrix (with dimensions of torque), taken in the present comparisons as

\[
\mathbf{A} = 2 \begin{pmatrix}
4250 & 0 & 0 \\
0 & 39,950 & 0 \\
0 & 0 & 39,950
\end{pmatrix} \quad \text{nt - m}
\]

\( \mathbf{b} \) is a control matrix (with dimensions of angular momentum):

\[
\mathbf{b} = \begin{pmatrix}
-3400 & 0 & 0 \\
0 & -8800 & 0 \\
0 & 0 & -8800
\end{pmatrix} \quad \text{kg - m}^2/\text{sec}
For the spacecraft under consideration in these comparisons, the moments of inertia in MKS units are:

\[
\mathbf{I} = \begin{pmatrix}
5,320 & 0 & 0 \\
0 & 88,950 & 0 \\
0 & 0 & 89,120
\end{pmatrix} \text{kg-m}^2
\]

with: \( GM = 3.98602 \times 10^{14} \text{ m}^3/\text{sec}^2 \)

\( R_0 = 6.6525535 \times 10^7 \text{ m} \) for a 90 minute circular orbital period.

Equations 1.4 and 1.5 can be expressed in vector form as an initial value problem as follows:

\[
\frac{d\mathbf{Y}}{dt} = \mathbf{F}(\mathbf{Y}, t)
\]

where:

\[
\mathbf{Y}(t_0) = \mathbf{Y}_0
\]

where:

\[
\mathbf{Y} = \begin{pmatrix}
q_1 \\
q_2 \\
q_3 \\
q_4
\end{pmatrix}
\]

\( t_0 \) is the initial time and
\[ \mathbf{y} = \begin{pmatrix} w_1(t_0) \\ w_2(t_0) \\ w_3(t_0) \\ q_1(t_0) \\ q_2(t_0) \\ q_3(t_0) \\ q_4(t_0) \end{pmatrix} \]
CHAPTER II

NUMERICAL EXPERIMENTATION PROCEDURE

The complexities of the system preclude an analytical solution, and hence as the true solution is unknown, the various numerical integration methods could only be compared with each other. The basis of comparison was the estimated truncation error as detailed below. No account of round-off error was made because this error depends on the computer used and the number of computations made in each method.

We assume that the local truncation error for a p-order single-step method has the form

\[ T(t, h) = g h^{p+1} \]  \hspace{1cm} (2.1)

where \( g \) is the principal error function (assumed essentially constant).

Then proceeding from \((t_{n-1}, Y_{n-1})\) to \((t_{n+1}, Y_{n+1})\), using two steps, each of size \( h \), the truncation error is approximately

\[ T(t_{n+1}, h) = g 2(h^{p+1}) \]  \hspace{1cm} (2.2)

so that

\[ Y(t_{n+1}) - Y^{(h)}_{n+1} = 2 g h^{p+1} \]  \hspace{1cm} (2.3)

With the same method, going from \((t_{n-1}, Y_{n-1})\) to \((t_{n+1}, Y_{n+1})\) in one step of size \( 2h \), the truncation error is

\[ T(t_{n+1}, h) = g(2h)^{p+1} \]  \hspace{1cm} (2.4)

\[ Y(t_{n+1}) - Y^{(2h)}_{n+1} = 2^{p+1} g h^{p+1} \]  \hspace{1cm} (2.5)

Subtracting 2.3 from 2.5 gives

\[ Y^{(h)}_{n+1} - Y^{(2h)}_{n+1} = g h^{p+1}[2^{p+1} - 2] \]  \hspace{1cm} (2.6)
Then approximately

\[ T(t_{n+1}, h) = \frac{y_{n+1} - y_{(2n+1)}}{2^{p+1} - 2} \]

Equation 2.7 was used to compare different methods. This is not an exact expression for the truncation error because \( g \) is not completely constant. Still this expression is a useful way to compare methods because it gives some measure of the truncation error involved. This estimate applies strictly to single-step methods, but was used for all methods since it still is some measure of the error and should tend to zero for any method as the step size decreases to zero even though it is not strictly to be interpreted as truncation error for multi-step methods.

Each method tested was run for the maximum step size, and then a number of runs were made with the step size halved successively. Equation 2.7 was used to estimate the error between the solution using step size \( h \), and the solution using \( h/2 \). This error was calculated at each time step and a root mean square of these errors was calculated. This RMS value was used to judge different methods.

The initial conditions used for the comparison were

\[ w(t_0) = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \quad q(t_0) = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \]

The period of orbit was 90 minutes. The maximum step size used was \( h = 0.001 \).

*All times and time steps without units are fractions of the orbit period.*
Figure 1 illustrates a typical solution up to $t = 0.25$ (quarter of an orbit).

With the initial conditions used, $w_1$, $w_2$, $q_1$, $q_2$, $q_4$ always remain unchanged, while $w_3$ and $q_3$ oscillate. This is to be expected as the gravity differential on the spacecraft is the only disturbing moment, and hence only $w_3$ and $q_3$ are affected. $q_3$ oscillates at twice the orbital period, and has no transient phase. $w_3$, however, has a transient phase, and dictates the step size, $h$, necessary for stability. For this reason, only the truncation error committed in $w_3$ was used to judge the methods tested. The error in $w_3$ by far outweighs the error in $q_3$.

Most methods were unstable for $h = 0.001$. With each method six more $h$'s were run, with $h$ halved for each run. Figure 1 shows that the fast oscillations of $w_3$ die down at about $t = 0.02$. The comparison scheme discussed above was used to calculate the RMS truncation error up to $t = 0.04$. This interval includes a period of fast oscillation, up to $t = 0.02$, and a period of slow periodic oscillation, up to $t = 0.04$. The $h = 0.001$ runs will obviously have the maximum truncation error, and each successive halving of $h$ will reduce this error because as $h \to 0$ the truncation error $\to 0$.

All runs were made in single precision arithmetic on a UNIVAC 1106 computer.
CHAPTER III

RESULTS OF COMPARISONS

Runge-Kutta Methods*

In Table I Runge-Kutta methods are compared by two parameters: RMS truncation error and computational time. The computational time is calculated as the number of function evaluations required by the particular method, i.e., the number of stages, $V$, for Runge-Kutta methods. Thus computation time is in units of number of function evaluations. Each method compared is identified by its name and its equation number in Appendix I.*

An X under a step size $h$, indicates that the method was unstable for that $h$. A - under a step size $h$, indicates that this step size was not run with this method. A 0 will occur for the smallest step size run because no truncation error can be calculated for the smallest step size.

Figure 2 shows examples of the errors in $w_3$ and $q_3$ for various step sizes plotted against time $t$. Figure 3 shows examples of the RMS $T$ plotted against step size for $w_3$. Both these figures are for the fourth-order Runge-Kutta-Ralston method.

Selecting a best method out of the ones tested is a judgment problem and depends on the users' requirements. Most users require both speed and accuracy. Hence the methods of Table I were judged on the basis of an error-time parameter, $ET$. This is a judgment

parameter which gives an equal weight to both the error and the computational time of a particular method:

\[ ET = \left( \frac{\text{RMS}}{2^{p+1} - 2} \right)^{\frac{1}{h}} \]

Based on this parameter, ET, the following observations are immediately made from Table I:

- Best fourth-order RKE: Ralston RKE(4,4)*
- Best fifth-order RKE: Butcher RKE(5,6)
- Best sixth-order RKE: Shanks RKE(6,6)
- Best seventh-order RKE: Sarafyan RKE(7,10)
  (except for \( h = 0.000125 \))
- Best eighth-order RKE: Shanks RKE(8,10)
  (except for \( h = 0.000125 \))

Now from among these five methods, the best choice, based on the ET parameter, is Shanks RKE(8,10) with \( h = 0.00025 \).

The results can be interpreted in another way: it is better, i.e., smaller ET, to use the Butcher RKE(5,6) at \( h = 0.000125 \) than to use the Shanks RKE(8,10) at \( h = 0.0005 \). Another observation, evident from Table I, is that the sixth, seventh, and eighth-order methods reach minimum ET's, but not at the smallest step size.

It is again emphasized that the judgment parameter ET gives equal weight to error and time.

**Multi-Step Methods**

Although a complete investigation of the multi-step methods is desirable, the available time permitted only a limited investigation.

*RKE(q,r) refers to an explicit Runge-Kutta method of q-order with r stages.
With the time factor in mind, three of the most promising types of predictors were chosen - Adams-Bashforth (AB), Krogh, and Craine-Klopfenstein (CK), and three types of correctors were chosen - Adams-Moulton (AM), Rodabough-Wesson (RW) and Wesson. Also, a Butcher fifth order hybrid method was chosen to compare with the multi-step methods.

The predictor and corrector equations were used to solve the initial value problem in various P-C combinations in both the PEC and PECE modes. Also the PE(CE)^S mode was run with an accelerated Jacobi scheme; however, the acceleration parameter was found to be zero for this solution. Thus, the PE(CE)^S mode was discarded. The result of the experimentation is presented in Table II.

From Table II, the best of the three types of predictors was the CK predictor, and the best of the three types of correctors was the AM corrector. To verify these conclusions the following observations were made:

1) The three predictors were used in P-C combinations with the sixth-order AM corrector. The fourth-order CK was found to give a truncation error 9.4 times less than that of the fourth-order Krogh predictor for a time step of .000125. Also, the fourth-order CK was found to give a truncation error 16.5 times less than that of the fourth-order AB predictor at the same time step.

2) The fifth-order Butcher hybrid gave 3 times less truncation error than the eighth-order AB-AM combination for a time step of .000125. However, the sixth-order CK-AM combination gave 2 times less truncation error than the fifth-order Butcher hybrid method.
3) The fourth-order CK-sixth-order AM combination gave 6.5 times less truncation error than the eighth-order AB-AM combination.

4) The RW correctors gave solutions that did not closely compare with the other Runge-Kutta and multi-step solutions. Thus, the RW correctors were discarded.

5) The fourth-order CK-eighth-order AM combination had slightly less truncation error than the fourth order CK-eighth-order Wesson combination for all timesteps considered.

Based on the merit factor, ET, introduced above, the fourth-order CK-eighth-order AM combination is the best of those considered.

The best multi-step methods considered are generally less effective than the best Runge-Kutta methods when judged by this merit factor, and stability limitations preclude the use of the larger step sizes allowed with the Runge-Kutta methods with the higher-order methods. It thus appears that the best Runge-Kutta methods are to be preferred for this system of differential equations.

Extrapolation Methods

Six extrapolation methods were investigated. These methods used rational function and polynomial function extrapolation with the Euler and modified midpoint algorithms. The Euler algorithm was used with polynomial function extrapolation with the basic time step being subdivided according to each of the sequences

\[ h_k = h_0/2^k \]

and

\[ h_k = \{h_0, \ h_0/2, \ h_0/3, \ h_0/4, \ \cdots \} \]

The modified midpoint rule was used with both rational function and polynomial function extrapolation with each of the above sequences.
All methods used local extrapolation. The A-stable trapezoidal rule was not used since it requires global extrapolation. Further details on the Euler-Romberg and Bulirsch-Stoer modified midpoint method can be found in Reference [556].

Since the extrapolation methods subdivide each time step repeatedly until a desired tolerance between successive extrapolations is obtained, comparisons with other methods is not directly possible. However, some conclusions can be drawn concerning the best of the extrapolation methods with regard to accuracy and number of function evaluations.

Extrapolation methods have as variable parameters the basic step size and the tolerance between successive extrapolations. In this work there is no automatic step size correction. Results for comparison purposes were obtained by making a number of computer runs at different basic step sizes with constant tolerance between successive extrapolations and then repeating the runs with successively smaller tolerances between successive extrapolations. These results indicate an optimum value of tolerance and step size for each method. As the basic step size increases, the number of extrapolations (function evaluations) needed to obtain a given tolerance decreases. However, for large step sizes the accuracy of the extrapolated values tends to decrease. The optimum values must be determined for each method experimentally. If the tolerance between successive extrapolations is too low, accuracy is poor, while if too high, instability can occur.

All midpoint rule methods became unstable at a basic step size of .00037 (fraction of one orbit), some at a smaller step size. So apparently extrapolation cannot totally overcome the instability
characteristic of the midpoint rules. When using a halving sequence, polynomial function extrapolation required more function evaluations than the rational function extrapolation with little difference in accuracy. If a reciprocal sequence was used the polynomial function extrapolation required fewer function evaluations than the rational function extrapolation method but gave less accurate results and went unstable at a lower step size. Thus there is little advantage to either rational function or polynomial function extrapolation, and the reciprocal and halving sequences produced no real savings in function evaluations when used with the midpoint rule.

Of the two Euler methods which both used polynomial function extrapolation, the solution obtained with the reciprocal sequence was very slightly more accurate than the solution obtained by successively halving the time step. The number of function evaluations was from 2 to 8 times fewer for the solution with the reciprocal sequence of step size subdivision, depending upon the smallness of the tolerance between successive extrapolations.

Partial results are presented in Figure 4. The best two methods are shown, these being the Euler algorithm with polynomial function extrapolation utilizing a reciprocal sequence for step size subdivision (ERPESR) and the modified midpoint rule with rational function extrapolation utilizing a halving sequence (BSRESH).

The accuracy of both methods is seen to be a function of step size and tolerance between successive extrapolations. BSRESH approaches the accuracy of ERPESR but requires about 40 times as many function evaluations at a step size of .000185 (fraction of one orbit). At a step size of .00037 the order of the function evaluations is the
same but BSRESH goes unstable at higher values of time. Extrapolation to higher tolerances increases accuracy, particularly at lower step sizes, at the expense of increased function evaluations for both BSRESH and ERPESH. However, there is a limit to the tolerances for both methods below which the solution becomes unstable. And, of course, if the tolerance is too large accuracy decreases.

ERPESH had constant accuracy over a large range of tolerances ($10^{-4} - 10^{-10}$) for step sizes between $0.0000231$ and $0.000185$. However, at larger step sizes the accuracy of the solution decreased at lower tolerances ($10^{-4}$).

Thus the best of the extrapolation methods is ERPESR, Euler algorithm with polynomial function extrapolation using a reciprocal sequence to subdivide the basic step size until a tolerance between successive extrapolation of $10^{-4}$ to $10^{-10}$ is achieved. This method gives better accuracy than the Runge-Kutta and predictor corrector methods and takes about the same number of function evaluations. A value of angular velocity obtained by the best of the predictor corrector methods is shown on Figure 5 at a step size of $0.000185$. This value is 1.4% lower than the value obtained by ERPESR and required 40 function evaluations as compared to 41 function evaluations for ERPESR when the extrapolation was carried to a tolerance of $10^{-4}$, or 71 function evaluation for a tolerance of $10^{-10}$. Thus ERPESR appears to require the same order of function evaluation as Runge-Kutta and predictor corrector methods at a better accuracy.
CHAPTER IV
SURVEY OF TECHNIQUES FOR FURTHER CONSIDERATION

The numerical solution of the initial value problem

\[ y' = f(t, y) \quad , \quad y(0) = y_0 \]  

has been approached in various ways as outlined below. As is noted, the various classes cited are not exclusive, but overlap one another.

**Quadrature Methods**

One may write formally

\[ y(t_n) = y(t_{n-1}) + \int_{t_{n-1}}^{t_n} f(t, y(t)) \, dt \]  

In the quadrature methods, the integral in 4.2 is approximated by a numerical quadrature expression. For instance, suppose it is desired to calculate the numerical approximation, \( y_n \), at a series of times, \( t_n \), \( n=0, 1, 2, \ldots \), called the nodes hereafter for convenience of notation, these nodes not necessarily being equally spaced. Then we may approximate the integral of 4.2 by a numerical quadrature consisting of a linear combination of nodal values and possibly also some intermediate values at points interspersed among the nodes. The coefficients in the linear combination and the locations of the nodes and/or the inter-nodal points are determined by the particular type of numerical quadrature employed. These factors are obtained directly from quadrature expressions for some types of methods, but by only indirect reference to numerical quadrature in others, the former type being that strictly referred to as "quadrature methods" in many works.
If only previously calculated nodal values, and possibly the value at the node presently under consideration, are included in the quadrature formula, then no additional equations are required, and the value at the node presently under consideration is determined either explicitly or implicitly through the solution of a nonlinear equation, depending on whether or not the value at the node under consideration is included in the quadrature expression. Methods of this type are also a sub-class of the linear multi-step methods discussed below.

Nesterchuk [378] gives such an implicit method involving all previous nodal points.

If nodal points beyond the node presently under consideration are included in the quadrature formula then an equation must be written for each of these points, and a system of equations must then be solved simultaneously for all the unknown nodal values involved. This type method is also a sub-class of the block linear multi-step methods discussed below.

Day [116] gives a method of this type using Lobatto quadrature and successively higher moments of the differential equation to supply the needed equations for all the unknown nodal values included in the block. For linear equations this method achieves \( n+1 \) order with \( n \) function evaluations and is reported to be superior in accuracy to some Runge-Kutta methods.

If inter-nodal points are involved, additional equations for the values at these points must be included. If these inter-nodal values are determined in turn by quadrature formulas, we have the standard Runge-Kutta methods. Within this class we have explicit methods if the expressions for the inter-nodal values involve only values at previous inter-nodal points, semi-implicit methods if these
expressions involve the presently considered inter-nodal value as well but none beyond, and implicit methods if inter-nodal values beyond that presently considered are included. The latter types require, respectively, the solution of one nonlinear equation at each internodal point or the simultaneous solution of a system of equations, one for each internodal point.

The method of Hulme [253] is a generalization of this implicit Runge-Kutta form constructed using Gaussian quadrature to approximate the integrals involved in a Galerkin approximation with the solution represented by a polynomial within each step. This produces a piecewise continuous solution with essentially the same effort required by conventional implicit Runge-Kutta methods and the same nodal values as produced thereby. Axelson [13], in a general discussion of quadrature methods, presents implicit Runge-Kutta methods based on Radau and Lobatto quadratures. Implicit Runge-Kutta methods are also discussed in Gourlay [194], where it is noted that certain methods stable for linear equations, i.e., with constant Jacobian matrices, may not be stable when the Jacobian varies as is the case with nonlinear equations. The common trapezoidal method is a case in point, and a stable modification of the same order is given therefor.

Explicit Runge-Kutta methods with provision for intrinsic truncation error estimation are given in Warten [536], Sarafyan [448], and Zonneveld [553]. Haines [220] gives a semi-implicit Runge-Kutta method with the coefficients chosen to increase stability and with the Jacobian matrix, required in such methods, evaluated by finite differences. Semi-implicit Runge-Kutta methods are also given by Allen [3].

Sarafyan [452] fits a Hermite polynomial to the inter-nodal values of an explicit embedded Runge-Kutta method to produce a continuous approximation of order only one less than that of the discrete approximation. Merson [356] gives Runge-Kutta methods for which each inter-nodal value is required to be a good approximation of the solution at the corresponding point. Treanor [519] uses local linearization before the quadrature to develop a modification of the Runge-Kutta type. The algebraically difficult derivation of the coefficients involved in Runge-Kutta methods is relegated to the computer via a program of Sarafyan and Brown [447]. Many other Runge-Kutta methods are discussed in Appendix I.
The more general inclusion of inter-nodal points along with nodal points in the quadrature expression for the integral of 4.2 results in a combination of the above types, variously referred to as multi-step Runge-Kutta methods or hybrid methods.

Rosen [430] gives an explicit multi-step Runge-Kutta method using at each node all the inter-nodal values used at the previous node, with a resultant decrease in the number of evaluations per step required for a given order.

Finally, in the manner already discussed above, the inclusion of nodal and/or inter-nodal points beyond the node presently under consideration produces block methods of the Runge-Kutta or hybrid type, requiring simultaneous solution for all the unknown nodal and/or inter-nodal values involved.

Block Runge-Kutta methods in which the iteration at each node is intentionally not carried to convergence are discussed by Rosser [433]. These methods require fewer evaluations per step for a given order than the usual point Runge-Kutta methods.

**Multi-value Methods**

At each nodal point, one may write, more generally, the value of the dependent variable, any of its derivatives, and any functions or combinations thereof as linear combinations of these quantities at each node and/or at inter-nodal points, and so produce the almost all-inclusive class of multi-value methods.

Methods of this type are discussed in general in Gear [177]. Methods with variable step size are discussed in regard to stability in Tu [520].

The entire class of quadrature methods discussed above is a subclass of the multi-value methods for which the value of the dependent variable, \( y \), at only one previous nodal point is involved, and the linear combination includes only the first derivative, \( y' = f(t, y) \).
The use of values of only the dependent variable and its first derivative in the linear combination produces the linear multi-step methods. These methods are explicit if only values of the derivative at previous nodes are included, semi-implicit (commonly called implicit) if the derivative at the presently-considered node is included as well but none beyond, and implicit (commonly called "block") if nodal values beyond that presently under consideration are included. The latter types require, respectively, the solution of a nonlinear equation at each node or the simultaneous solution of a system of equations for all the unknown nodal values involved. Again inter-nodal points may be included as well, and the so-called hybrid methods developed as discussed above. The more common types of predictor-corrector methods, in which values are "predicted" by explicit multi-value forms and then "corrected" using implicit multi-value forms involving the predicted value as well as previously calculated values are in this class. Additional applications of the corrector may follow, using the most recently corrected values.

Hull and Creemer [244], in a comparison of various predictor-corrector forms, state that error, stability, and the order required for a given error at least computation are less sensitive to order with two corrector applications than with one, with no significant improvement thereafter. Lambert [308] shows the equivalence of any predictor-corrector method using a finite number of corrector applications to some explicit multi-step method. This limits the expectations for stability of predictor-corrector methods. Donelson and Hansen [129] use a set of correctors applied cyclically over a set of time steps and thus achieve higher-order stable methods than are possible with the same corrector used at all steps. Order 2k-1 is thereby achieved for stable k-step methods, as opposed to the maximum order, k-1, possible with stability for methods using only a fixed corrector. The use of variable step size in predictor-corrector methods has been considered by Van Wyk [532], and in a single-step method of the multi-step form by Richards, Lanning, and Torrey [419]. Other predictor-corrector methods are given in Appendix II.

Many multi-step methods have been developed in which some coefficients are chosen to increase stability as in Krough [291],
with a comparison of stability plots for various methods; Crane and Lambert [96]; Nigro [383]; Rahme [406], with an attempt to minimize error while maintaining stability; and Lomax [332], considering Runge-Kutta and hybrid methods as well. A multi-step method with the coefficients chosen to fit the characteristic roots to some of the characteristic exponentials of the differential system is given by Miranker [366] for non-linear systems with local linearization, the method being implicit, and for linear systems by Liniger and Willoughby [323], an explicit method. Osborne [398] gives a single-step method with the nodes chosen to give desired characteristic roots. Timlake [518] increases the stability of multi-step methods by averaging over previous steps, determining the weights from the Jacobian eigenvalues. The stability of multi-step methods has been analyzed particularly in Karim [258,256], sufficient conditions for instability being given in the former and the effect of the predictor as well being included in the latter; Dahlquist [100-106], a series of classic papers; and Hafner [217], giving stability charts for a large number of Runge-Kutta and hybrid methods as well.

Implicit multi-step methods particularly for stiff systems have been given by Ratliff [411], giving a comparison of several methods; Jain and Srivastava [246,247]; Dill [126], a systematic search for such methods; Gelinas [182]; Dill and Gear [125], again a search; and Gear [172]. Such methods achieve a particular shape of stability boundary suitable for stiff systems by always including the derivative at the present node with perhaps the derivatives at a few previous nodes.

Tyson [521] gives an implicit multi-step method for nonlinear systems using local linearization about a predicted value, rather than the previous nodal value, that achieves order two greater than that of the predictor used. (Local linearization is used in many methods to render nonlinear systems tractable by methods restricted to linear systems, but such linearization about the previous nodal value restricts the order to two regardless of the method used.) Boggs [30] used Broyden iteration (involving direct approximation of the Jacobian inverse rather than inversion of the Jacobian) with implicit multi-step methods. Block methods are given by Daniel [109,108] and by Shampine and Watts [469]. Methods for determining the necessary set of starting values (a problem equivalent to the development of block multi-step methods) have been given in Reimer [412], Rakitskii [407], and Alonzo [4].

The addition of inter-nodal points (hybrid methods) has been considered by Papian and Ball [399]; Lomax [332], who gives a method for choosing the coefficients in explicit methods of this type to improve stability; Gragg and Stetter
discussing the coefficients therein for maximum order; Gear [175]; and Hafner [217], who gives a large number of stability charts. Additional consideration of multi-step methods has been given by Spijker [492], considering the question of direct application to higher-order equations or a split to a system of first-order equations, and by Zverskina [555], using derivatives only at previous nodes separated by some distance from the present node.

Various other quantities, such as higher derivatives and divided differences, have been included in the linear combination, and methods using one type of quantities may be derived by transformation of methods using another type as in Osborne [395] and Kohfeld and Thompson [283]. Again the above-mentioned ramifications involving inter-nodal points and/or nodal points beyond that presently under consideration may be developed. A further modification is that values at previously passed nodes may be subsequently changed. It is possible to derive methods of this type that use higher derivatives or higher-order divided differences, all of which are produced at each node from linear combinations of the same, in lieu of values at previous nodes. In this manner more efficient step size change can be accomplished.

Non-Polynomial Interpolates

In the great majority of methods the linear combinations involved in the multi-value methods, and thus in the quadrature methods, are required to be exact when the dependent variable is a polynomial of some degree. This restriction may be relaxed, and an even broader spectrum of methods may be developed by requiring these linear combinations to be exact for some other function, in particular for perhaps some class of functions especially adapted to the particular differential equation being considered.

Loscalzo and Talbot [339] use a spline function with the higher derivatives matched at the nodes, which, while producing the same nodal values developed by a multi-step method, is a single-step method and gives a piecewise continuous approximation. It does, however, require differentiation. Byrne and Chi [45] use a spline approximation of the integrand in a quadrature type method requiring no differentiation. Callender [71] uses a spline approximation spanning several nodes to develop a block multi-step method.

Blue and Gummel [27] and Lambert and Shaw [306] use rational function approximation of matrix exponentials, the former using coefficients chosen to increase stability and producing thereby Pade approximates of the exponential. The latter method matches higher derivatives and thus requires differentiation. Roe [84] gives a multi-step method using a combination of a polynomial and an exponential as the interpolate. Other non-polynomial interpolates have been used by Shaw [475], Lambert and Shaw [303], and Pope [403], the first two being primarily of use when singularies are involved in the solution.

Only a relatively small number of methods of this type have been considered, since the intention of most developments has been to develop methods that may be applied at least fairly well to all types of differential equations, hence the use of polynomial interpolation. In this connection the remarks of Lomax [334] are pertinent:

"It is unlikely that 'new' combinations of linear equations that connect a function, u, and its derivative, u', at a series of reference points, equispaced or not,"
will improve existing methods for the numerical integration of general sets of coupled ordinary differential equations..." (p. 2). "What appears to be needed are studies of special methods designed for special methods designed for special classes of equations..." (p. 6).

**Iterative Methods**

Methods of this class start with some approximate solution, such as the solution of a related but simpler differential equation or the result of any numerical method, and improve the approximation by an iterative procedure. One such method is the Lie series method. The extrapolation methods by which solutions with successively smaller truncation errors are obtained by extrapolating results with successively smaller step-sizes may also be considered iterative in the sense stated.

In reference [207] Runge-Kutta methods having error expansions in even powers of the step size are developed to serve as the basis for these extrapolation methods. Lie Series methods have been given by Knapp and Wanner [279,280].

**Transformation Methods**

In methods of this type the dependent and/or independent variable is transformed before numerical solution in order to obtain a form with better numerical properties. In this manner stiff systems may be transformed into systems with smaller ranges of eigenvalues.

Lawson [313] gives a transformation of the dependent variable which reduces the stiffness of the system, but the method requires the use of matrix exponentials. A similar transformation of the dependent variable is used by Jain [259]. Other transformations of the dependent variable are given by Decell, Guseman and Lea [117], requiring evaluation of the Jacobian inverse, and by Calahan [67].

**Higher-Order Equations**

As is well known, any higher-order differential equation can be broken into a set of simultaneous first-order equations to which the
preceding methods are applicable. Generalizations of the classes cited follow readily, however, so that application may be made directly to the higher-order equation if such is thought to be preferable. This is still something of an open question.

Osborne [397] gives a quadrature method for linear higher-order equations. Day [112,113] has implicit Runge-Kutta type methods using Gaussian and Lobatto quadrature, respectively, for second-order equations with no first derivative. Cooper [91,93] gives a generalization of the implicit Runge-Kutta methods to equations of any order and states that the split into a set of first-order equations decreases the local accuracy and increases the computation required, with little effect on the global accuracy.

The multi-value methods using higher derivatives are directly applicable to higher-order equations, and Gear [177] states also that direct solution of the higher-order equations may be faster than that of the split system of first-order equations. Methods of this type are also given in Gear [178, 484] and Allen [2]. Adrianova [1] found less error without splitting to the first-order system. However, Spijker [503] states that round-off error is reduced by splitting into first-order equations.

Conclusions and Directions

Implicit methods in general are much more stable than explicit methods, but may still be less efficient because of the difficulty of the solution of the system of simultaneous equations that is required at each time step. The key to the use of the more stable implicit schemes is thus the iterative solution of this system, and effort should be directed toward the development of faster and more appropriate iterative schemes for use in these methods. There are, of course, a great many general iterative schemes that have never been applied in implicit numerical solutions of ordinary differential equations.

Block methods with a set number of iterations, such as that of Rosser [432], can achieve a given order with fewer evaluations per step
than for point methods and are thus attractive. The number of evaluations required may also be decreased by the retention of previously calculated inter-nodal values as in the method of Rosen [430].

The many possible forms of the multi-value methods of Gear [177], in particular those that update previously calculated values, may offer more efficient methods than the conventional multi-step forms. Stable methods of higher order than possible for multi-step methods are attainable in this class. Methods of this type may be developed from the existing multi-step methods by linear transformations, and yet may be more efficient in computation.

The local linearization of nonlinear systems about a predicted value, Tyson [521], rather than the value at the previous step, allows linear methods to be applied without limiting the resultant overall order for the nonlinear system to two. This procedure has received little attention but could widen considerably the scope of application of many methods applicable only to linear systems.

The use of a cyclic set of correctors, Donelson and Hansen [129], rather than a fixed corrector, over a series of time steps yields stable methods of higher order than possible with a fixed corrector and this should be pursued.

Methods using non-polynomial interpolates specialized for particular systems of differential equations should be considered. Here speed may quite possibly be gained from the loss of generality. In the same respect, preliminary transformations of the dependent variable adapted to the particular differential system should also be considered.
<table>
<thead>
<tr>
<th>Explicit Runge-Kutta</th>
<th>RK(p,V)</th>
<th>EQ. IN AP-PEN-DIX</th>
<th>V T I M E</th>
<th>( \text{RMS TE} \times 10^{-10} )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>( h = \text{(RMS TE/2}^{1/2}(V/h) \times 10^6 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>( h = 0.001 )</td>
</tr>
<tr>
<td>Classic(6,4)</td>
<td>A.26</td>
<td>4</td>
<td>X</td>
<td>1.578x10^4</td>
</tr>
<tr>
<td>Dill(4,4)</td>
<td>A.25</td>
<td>4</td>
<td>X</td>
<td>4.208x10^4</td>
</tr>
<tr>
<td>Ralston(4,4)</td>
<td>A.21</td>
<td>4</td>
<td>X</td>
<td>1.565x10^4</td>
</tr>
<tr>
<td>Nystrom(5,6)</td>
<td>A.34</td>
<td>6</td>
<td>X</td>
<td>1.006x10^4</td>
</tr>
<tr>
<td>Butcher(5,6)</td>
<td>A.38</td>
<td>6</td>
<td>X</td>
<td>1.107x10^3</td>
</tr>
<tr>
<td>Lawson(5,6)</td>
<td>A.42</td>
<td>6</td>
<td>X</td>
<td>2.973x10^8</td>
</tr>
<tr>
<td>Fehlberg(5,6)</td>
<td>A.49</td>
<td>6</td>
<td>X</td>
<td>4.64x10^6</td>
</tr>
<tr>
<td>Luther(5,6)</td>
<td>A.35</td>
<td>6</td>
<td>X</td>
<td>3.65x10^4</td>
</tr>
<tr>
<td>Gauss Quadrature</td>
<td>A.43</td>
<td>6</td>
<td>X</td>
<td>2.88x10^4</td>
</tr>
<tr>
<td>Luther(5,6)</td>
<td>A.44</td>
<td>6</td>
<td>X</td>
<td>1.02x10^3</td>
</tr>
<tr>
<td>Radau Quadrature</td>
<td>A.45</td>
<td>6</td>
<td>X</td>
<td>1.02x10^3</td>
</tr>
<tr>
<td>Shanks(5,5) N=3</td>
<td>A.47</td>
<td>5</td>
<td>X</td>
<td>9.898x10^11</td>
</tr>
<tr>
<td>Shanks(5,5) N=100</td>
<td>A.47</td>
<td>5</td>
<td>X</td>
<td>9.492x10^11</td>
</tr>
<tr>
<td>Modified Shanks(5,4) N=100</td>
<td>A.48</td>
<td>4</td>
<td>X</td>
<td>2.64x10^6</td>
</tr>
<tr>
<td>Butcher(6,7)</td>
<td>A.57</td>
<td>7</td>
<td>X</td>
<td>1.046x10^10</td>
</tr>
<tr>
<td>Newton-Cotes Quadrature</td>
<td>A.67</td>
<td>6</td>
<td>X</td>
<td>1.333x10^12</td>
</tr>
<tr>
<td>Shanks(6,6)</td>
<td>A.67</td>
<td>6</td>
<td>X</td>
<td>4.57x10^3</td>
</tr>
<tr>
<td>Sarafyan(7,10)</td>
<td>A.68</td>
<td>10</td>
<td>X</td>
<td>5.87x10^3</td>
</tr>
<tr>
<td>Shanks(7,7)</td>
<td>A.73</td>
<td>7</td>
<td>X</td>
<td>5.16x10^2</td>
</tr>
<tr>
<td>Curtis(8,11)</td>
<td>A.75</td>
<td>11</td>
<td>X</td>
<td>3.10x10^6</td>
</tr>
<tr>
<td>Shanks(8,10)</td>
<td>A.77</td>
<td>10</td>
<td>X</td>
<td>2.94x10^6</td>
</tr>
<tr>
<td>Shanks(8,10)</td>
<td>A.77</td>
<td>10</td>
<td>X</td>
<td>5.78x10^5</td>
</tr>
</tbody>
</table>

The signed number following each entry indicates multiplication by the corresponding power of 10.
<table>
<thead>
<tr>
<th>MULTISTEP METHOD</th>
<th>EQ. IN APPENDIX</th>
<th>V TIME</th>
<th>RMS TE $10^{-10}$ ET = ($RMS~TE/2^{P+1-2}$)(V/h) $10^x$</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB(3, 3)</td>
<td>31</td>
<td>X</td>
<td>1.54 x 10^4 1.955 x 10^3 0</td>
</tr>
<tr>
<td>AN(5, 2) (PECE)</td>
<td>55</td>
<td></td>
<td>8.80 x 6 2.23 x 6</td>
</tr>
<tr>
<td>AB(3, 3)</td>
<td>31</td>
<td>X</td>
<td>6.75 x 10^3 1.185 x 10^3 0</td>
</tr>
<tr>
<td>AN(4, 3) (PECE)</td>
<td>56</td>
<td></td>
<td>1.80 x 6 6.32 x 7</td>
</tr>
<tr>
<td>AB(3, 3)</td>
<td>31</td>
<td>X</td>
<td>7.77 x 10^3 1.07 x 10^3 0</td>
</tr>
<tr>
<td>AN(4, 3) (PEC)</td>
<td>56</td>
<td></td>
<td>1.03 x 6 2.85 x 7</td>
</tr>
<tr>
<td>CK(4, 4)</td>
<td>4</td>
<td>X</td>
<td>- 4.24 x 10^2 40.6 0</td>
</tr>
<tr>
<td>AN(4, 3) (PECE)</td>
<td>56</td>
<td></td>
<td>2.26 x 7 4.3 x 8</td>
</tr>
<tr>
<td>Butcher Hybrid</td>
<td>1</td>
<td>X</td>
<td>1.71 x 10^3 57.2 0</td>
</tr>
<tr>
<td>Fifth order</td>
<td></td>
<td></td>
<td>3.30 x 7 2.22 x 8</td>
</tr>
<tr>
<td>CK(4, 4)</td>
<td>4</td>
<td>X</td>
<td>4.12 x 10^3 1.55 x 10^2 31.7 0</td>
</tr>
<tr>
<td>Wesson(5, 5) (PECE)</td>
<td>73</td>
<td></td>
<td>5.32 x 7 4.0 x 8 1.5 x 8</td>
</tr>
<tr>
<td>AB(4, 4)</td>
<td>32</td>
<td>X</td>
<td>2.57 x 10^3 6.09 x 10^2 0</td>
</tr>
<tr>
<td>AN(6, 5) (PEC)</td>
<td>58</td>
<td></td>
<td>5.16 x 7 1.75 x 7</td>
</tr>
<tr>
<td>Krogh(4, 4)</td>
<td>25</td>
<td>X</td>
<td>2.61 x 10^3 3.46 x 10^2 0</td>
</tr>
<tr>
<td>AN(6, 5) (PECE)</td>
<td>58</td>
<td></td>
<td>5.23 x 7 9.85 x 8</td>
</tr>
<tr>
<td>AB(4, 4)</td>
<td>32</td>
<td>X</td>
<td>1.40 x 10^4 2.46 x 10^3 0</td>
</tr>
<tr>
<td>AN(6, 5) (PEC)</td>
<td>58</td>
<td></td>
<td>1.40 x 6 3.53 x 7</td>
</tr>
<tr>
<td>Krogh(4, 4)</td>
<td>25</td>
<td>X</td>
<td>1.40 x 10^4 2.46 x 10^3 0</td>
</tr>
<tr>
<td>AN(6, 5) (PECE)</td>
<td>58</td>
<td></td>
<td>1.40 x 6 3.53 x 7</td>
</tr>
<tr>
<td>CK(4, 4)</td>
<td>4</td>
<td>X</td>
<td>1.537 x 10^3 82.6 0</td>
</tr>
<tr>
<td>AN(6, 5) (PECE)</td>
<td>58</td>
<td></td>
<td>9.77 x 8 1.05 x 8</td>
</tr>
<tr>
<td>CK(4, 4)</td>
<td>4</td>
<td>X</td>
<td>4.81 x 10^3 3.13 x 10^2 0</td>
</tr>
<tr>
<td>AN(6, 5) (PEC)</td>
<td>58</td>
<td></td>
<td>1.55 x 7 1.99 x 8</td>
</tr>
<tr>
<td>AB(8, 8)</td>
<td>36</td>
<td>X</td>
<td>- 2.1 x 10^3 7.9 x 10^2 0</td>
</tr>
<tr>
<td>AN(8, 7) (PECE)</td>
<td>60</td>
<td></td>
<td>6.58 x 8 4.97 x 8</td>
</tr>
<tr>
<td>CK(4, 4)</td>
<td>4</td>
<td>X</td>
<td>- 85 22.8 0</td>
</tr>
<tr>
<td>AN(8, 7) (PECE)</td>
<td>60</td>
<td></td>
<td>2.0 x 9 1.43 x 10</td>
</tr>
<tr>
<td>CK(4, 4)</td>
<td>4</td>
<td>X</td>
<td>- 96.5 24 0</td>
</tr>
<tr>
<td>Wesson(8, 8) (PECE)</td>
<td>74</td>
<td></td>
<td>3.3 x 9 1.5 x 10</td>
</tr>
</tbody>
</table>
Figure 1. $W_3$ and $q_3$ up to $T = 0.25$ (quarter of an orbit).
Figure 2. Errors in $W_3$ and $q_3$ versus time, up to $T = .04$. 
MULTIPLE OF BASIC TIME STEP

Figure 3. RMS error in \( W_3 \) versus step size.
Figure 4. Comparison of ERPESR and BSRESH Extrapolation Methods - Time Step Effects.
Figure 5. Comparison of ERPESR and BSRESH Extrapolation Methods - Tolerance Effects.


120. Dejon, B., "Addendum to Stronger Than Uniform Convergence Multi-

121. Dennis, S., "The Numerical Integration of Ordinary Differential
Equations Possessing Exponential Type Solutions," Proc.

122. Dennis, S., "Step By Step Integration of Ordinary Differential

123. Descloux, J., "A Note on a Paper by A. Nordsieck," Dept. of

124. Devyotko, V., "On a 2-Sided Approximation for the Numerical
Integration of Ordinary Differential Equations," U.S.S.R.

125. Dill, C., & Gear, C., "A Graphical Search for Stiffly Stable
Comp. Mach. 18, 75 (1971).

Methods for Ordinary Differential Equations," Dept. of


128. Dolgopolova, T., & Ivanov, V., "On Numerical Differentiation,"

129. Donelson, J., & Hansen, E., "Cyclic Composite Multistep Pre-

130. Durham, H., et al., "Study of Methods for Numerical Solution of


Solution of Systems of Differential Equations," BIT 8, 276
(1968).

133. Ehle, B., "On Pade Approximations to the Exponential Functions
and A-Stable Methods for the Numerical Solution of Initial
Value Problems," Res. Rept. CSRR 2010, University of

134. Eisenpress, H., & Bomberault, A., "Efficient Symbolic Differenti-
tation Using PL/1-FORMAC," IBM New York Scientific Center


Simulation of Control Systems." CAL Report No. ZA-1681-E-1
(1964).

Comp. 20, 392 (1966).


167. Gaskill, R., Harris, J., & Knight, A., "DAS--A Digital Analog

168. Gaskill, R., "Fact & Fallacy in Digital Simulation," Sim. 5, 309
(1965).

169. Gates, L., "Numerical Solution of Differential Equations by Re-

170. Gautschi, W., "Numerical Integration of Ordinary Differential

171. Gear, C., "The Automatic Integration of Large Systems of Ordinary
Differential Equations," Digest Record of 1969 Joint Confe-
rence on Mathematical & Computational Aids to Design, Anaheim,

172. Gear, C., "The Automatic Integration of Stiff Ordinary Differen-
tial Equations," Information Processing 68, ed. A. J. H.

173. Gear, C., "The Automatic Integration of Ordinary Differential


175. Gear, C., "Hybrids Methods for Initial Value Problems in Ordinary

176. Gear, C., "Numerical Integration of Stiff Ordinary Differential
Equations," Report #221, University of Illinois, Dept. of
Computer Science (1967).

177. Gear, C., "The Numerical Integration of Ordinary Differential


APPENDIX I

RUNGE-KUTTA METHODS

This appendix is intended as a reference for various Runge-Kutta methods available in current literature.

Short form notation will be used henceforth: an Explicit p order, V stage Runge-Kutta method is written as RKE(p,V).

\[ \text{RKS} \text{I}(p,V) - \text{semi-implicit} \]
\[ \text{RK} \text{I}(p,V) - \text{implicit} \]
\[ \text{RKMS}(p,V) - \text{multi-step} \]

The Runge-Kutta coefficients will be presented in the following form:

\[
\begin{array}{c|c|c}
\hline
\text{c} & \text{a} & \text{T} \\
\hline
\w & \text{w} & \text{M} \\
\hline
\end{array}
\]

\[ k_i = hf(T_n + c_i h, Y_n + \sum_{i=1}^{M_i} a_{ij} k_j) \]

\[ Y_{n+1} = Y_n + \sum_{i=1}^{V} w_i k_i \quad i = 1, 2, \ldots, V \]

\[ M_i \leq V \]

In the following sections various references, where different Runge-Kutta methods may be found, are listed.

Many authors have developed the equations leading to the derivation of the coefficients defining a particular order Runge-Kutta method. These equations will be henceforth called the deriving equations; and these authors will be listed first. If a reader desires to develop his own Runge-Kutta method, he can do so by solving the
deriving equations. The field is still wide open, many Runge-Kutta methods can still be developed.

A.1 Explicit Runge-Kutta Methods

RKE(p,V):

Restating the equations for a RKE(p,V)

\[ k_i = hf[T_n + c_i h, Y_n + \sum_{j=1}^{i-1} a_{ij} k_j] \]

with \( c_1 = 0 \)

\[ Y_{n+1} = Y_n + \sum_{i=1}^{V} w_i k_i \]

The deriving equations for a RKE(p,V), based on a Taylor series analysis, (as developed in Ch. 2) are listed in; Butcher [3], up to a 8th order method; and Fehlberg [4, 12], up to a 8th order method.

A formulation for solving the non-linear deriving equations, for a RKE, on a computer is being developed and the principle is listed in Sarafyan and Brown [7].

The tedious Taylor series analysis for the deriving equations can be replaced by a, practically, equally tedious 'Quadrative Method.' The Quadrative Method for the deriving equations is, however, a more convenient form for use, as it is general for any older method, unlike the Taylor series form. Quadrative form deriving equations for any order RKE's are listed in Rosen [5].

An excellent reference for the general Runge-Kutta class of methods, all in one handy cover, is Lapidus and Seinfeld [6]. This book also lists commonly used Runge-Kutta methods of various orders.
References for RKE of specific orders are now listed. References for deriving equations will be given first, these will be the deriving equations of that specific order method.

A.1-1 RKE(1,1):

The famous (infamous?) Euler method may be considered to be such a method.

\[
\begin{array}{c|c|c}
0 & 0 & \text{Euler} \\
\hline
1 & & A.3
\end{array}
\]

A.1-2 RKE(2,2):

The deriving equations, specifically for a RKE(2,2) are listed in Ralston [8] and Fehlberg [12]. In Ralston [8] the free parameter was manipulated to give a Minimum Truncation Error Bound, henceforth referred to as TEB. It must be noted here that, the truncation error is not minimized but rather its bound is minimized. This may or may not result in a minimized truncation error. However, as the exact value of the truncation error depends on the differential equation being integrated, this course is the only one open for optimizing a method. Bounds of these kinds are quite conservative.

Lapidus and Seinfeld [6] list 3 such methods. One of them, the Heun form, is Ralston's optimum.

\[
\begin{array}{c|c|c}
0 & 2/3 & \text{Heun} \\
\hline
2/3 & 2/3 & \text{(optimum)} \\
\hline
1/4 & 3/4 & A.4
\end{array}
\]
When \( \frac{dy}{dt} \) is independent of \( Y \), the Heun form becomes a 3rd order method. Johnston [9], Kuntzmann[10], and King [11] also find optimum methods, these are the Heun form, eq. A.4.

Fehlberg [12] has developed methods of order \( p \), coupled with a \( p+1 \) order method. Suitable choice of free parameters is made so as to minimize the leading term of the truncation error. This results in a larger permissible step size. Comparison of the solution of the \( p+1 \) order with the \( p \) order is used to control step size.

The \( p+1 \) order shares most of the coefficients of the \( p \) order; hence with only 1 or 2 additional stages, a \( p+1 \) order solution is available. Though the Fehlberg methods require more stages than conventionally used; this extra calculation results in stepsize control, which is well worth the cost.

The automatic step size control feature of these Fehlberg methods make them computationally more efficient (speed and overall number of function evaluations), than conventional methods. This is so; because when step size controls, like the Richardson Extrapolation procedure (see Lapidus and Seinfeld [6]), are applied to conventional methods; the computational effort doubles. In the opinion of the author; these methods when used without the coupled
p+1 order method, are still useful because the TEB has been carefully minimized. The p+1 order method can also be used by itself.

Henceforth Fehlberg methods will be written as: \( \text{RKE}[(p,V); (p+1, V+n)] \). 

Fehlberg [12] RKE[(2,2); (3,3)]:

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>p=2</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>p=3</td>
<td>1/2</td>
<td>1/4</td>
<td>1/4</td>
<td></td>
</tr>
</tbody>
</table>

1/2 1/2 - w's for p=2

1/6 1/6 2/3 - w's for p=3

0

1/4 1/4

p=2 27/40 -189/800 729/800

p=3 1 214/891 1/33 650/891

214/891 1/33 650/891 p=2

533/2106 0 800/1053 -1/78 p=3

A.1-3 RKE(3,3):

The deriving equations for such methods are listed in Fehlberg [12], and in Ralston [8], as a two parameter family. In Ralston [8] the TEB was minimized by suitable choices of the two free parameters, and the 'optimum' method is:

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1/2</th>
<th>1/2</th>
</tr>
</thead>
<tbody>
<tr>
<td>3/4</td>
<td>0</td>
<td>3/4</td>
<td></td>
</tr>
</tbody>
</table>

2/9 1/3 4/9

Ralston Optimum

The two optimum methods of King [11] are, however, of a fourth and fifth order when $\frac{dY}{dT}$ is independent of $Y$. When $\frac{dY}{dT}$ is dependent on $Y$ his methods have a slightly larger TEB than Ralston's method (the method which becomes fourth order has a slightly smaller TEB than the method which becomes fifth order.)

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1/3</td>
</tr>
<tr>
<td>1/3</td>
<td>1/3</td>
</tr>
<tr>
<td>5/6</td>
<td>-5/12</td>
</tr>
<tr>
<td>1/10</td>
<td>1/2</td>
</tr>
</tbody>
</table>

King optimum (when $\frac{dY}{dT}$ independent of $Y$—fourth order)

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>.3550510257</td>
</tr>
<tr>
<td>.3550510257</td>
<td>.3550510257</td>
</tr>
<tr>
<td>.8449489743</td>
<td>-.4021612205</td>
</tr>
<tr>
<td>.1111111111</td>
<td>.5124858262</td>
</tr>
</tbody>
</table>

King optimum (when $\frac{dY}{dT}$ independent of $Y$—fifth order)

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(10-2\sqrt{13})</td>
</tr>
<tr>
<td>(10-2\sqrt{13})</td>
<td>6</td>
</tr>
<tr>
<td>.0581020</td>
<td>.8256939</td>
</tr>
<tr>
<td>.2071768</td>
<td>.3585646</td>
</tr>
</tbody>
</table>

Kuntzmann optimum

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(1+\sqrt{13})</td>
</tr>
<tr>
<td>(1+\sqrt{13})</td>
<td>6</td>
</tr>
<tr>
<td>A.11</td>
<td>6 A.12</td>
</tr>
</tbody>
</table>
Lapidus and Seinfeld [6] list three methods:

<table>
<thead>
<tr>
<th></th>
<th>Classic</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/2</td>
<td>1/2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>-1</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1/6 2/3 1/6</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Heun</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/3</td>
<td>1/3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2/3</td>
<td>0</td>
<td>2/3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1/4 0 3/4</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Nystrom</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2/3</td>
<td>2/3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2/3</td>
<td>0</td>
<td>2/3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1/4 3/8 3/8</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


Fehlberg [12] RKE[(3,4); (4,5)]:

<table>
<thead>
<tr>
<th></th>
<th>Fehlberg</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/4</td>
<td>1/4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4/9</td>
<td>4/81</td>
<td>32/81</td>
<td></td>
<td></td>
</tr>
<tr>
<td>p=3</td>
<td>6/7 57/98 -432/343 1053/686</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>p=4</td>
<td>1/6 0 27/52 49/156</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1/6 0 27/52 49/156</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>p=3</td>
<td>43/288 0 243/614 343/1872 1/12</td>
<td>p=4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The Fehlberg coupled second order methods, eqs. A.7 and A.8, could be used as third order methods.

The author developed optimum methods of a special kind, these methods require only two, instead of three function evaluations per step.

I. Back Step Method:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>24/100</td>
<td>24/100</td>
</tr>
<tr>
<td>76/100</td>
<td>19627/10200</td>
</tr>
<tr>
<td>409/684</td>
<td>-7/36</td>
</tr>
</tbody>
</table>

Butcher [18] developed such fifth order processes, using negative c's. This back step method has a built-in advantage: in every step except the first; \( k_3 \) from \( T_n, l/n \) may be used as the \( k_2 \) for \( T_{n+1}, Y_n \). This results in one less function evaluation per step, hence has \( V = 2 \) rather than \( V = 3 \), a considerable advantage. A disadvantage of back step methods is that a starting method is required. For a starting method solve the deriving equations with \( c_3 = \frac{76}{100} \) and any choice of \( c_2 \).
II. Front Step Methods:

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>A.19</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>1/10</td>
<td></td>
</tr>
<tr>
<td>1/10</td>
<td>1/10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 1/10</td>
<td>-4 133/170, 5 15/17</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>-1 5/22, 2 1/6, 17/66</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1/100</td>
<td>1/100</td>
<td></td>
</tr>
<tr>
<td>101/100</td>
<td>-990103/19700, 10000/197</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>-9994/606, 103/6, 197/606</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

These front step methods have a built-in advantage: in every step except the last, $k_3$ from $(T_n, Y_n)$ may be used as the $k_2$ for $(T_{n+1}, Y_{n+1})$. Like the back step method, this results in a $V = 2$ rather than $V = 3$ method. The front step method is self-starting and so has an advantage over the back step method.

Each of the methods eqs. A.18, A.19 and A.20 was optimized using the procedure found in Ralston [8]. The $c_2$'s and $c_3$'s were interchanged: a TEB was calculated using a particular $c_2$ and $c_3$; then another TEB was calculated using the same $c_2$ as a $c_3$, and the same $c_3$ as a $c_2$. Eqs. A.18, A.19 and A.20 are the resulting optimum methods.

<table>
<thead>
<tr>
<th>Method (p, V)</th>
<th>Error Bound (TEB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ralston A.9 (3,3)</td>
<td>0.1111 $ML^3h^4$</td>
</tr>
<tr>
<td>King A.10 (3,3)</td>
<td>0.1389 $ML^3h^4$</td>
</tr>
<tr>
<td>King A.11 (3,3)</td>
<td>0.1391 $ML^3h^4$</td>
</tr>
<tr>
<td>Classis A.13 (3,3)</td>
<td>0.5 $ML^3h^4$</td>
</tr>
<tr>
<td>Heun A.14 (3,3)</td>
<td>0.6667 $ML^3h^4$</td>
</tr>
<tr>
<td>Nystrom A.15 (3,3)</td>
<td>0.25 $ML^3h^4$</td>
</tr>
<tr>
<td>Back step A.18 (3,2)</td>
<td>0.3493 $ML^3h^4$</td>
</tr>
<tr>
<td>Front step A.19 (3,2)</td>
<td>0.3933 $ML^3h^4$</td>
</tr>
<tr>
<td>Front step A.19 (3,2)</td>
<td>0.3649 $ML^3h^4$</td>
</tr>
</tbody>
</table>
More has been written and analyzed about the RKE(4,4) than any other Runge-Kutta method. The references listed here try to sample all the aspects of these methods.

The deriving equations are listed in Butcher [3], Rosen [5], Ralston [8], and Fehlberg [12].


\[
\begin{array}{cccc|c}
0 & .4 & .4 & \text{Ralston Optimum} \\
.45573725 & .29697761 & .15875964 & \\
1 & .21810040 & -3.05096516 & 3.83286476 \\
\hline
.17476028 & -.55148066 & 1.20553560 & .17118478
\end{array}
\]

In King [11], two optimum methods are derived, both whose TEB's are larger than Ralston's optimum, eq. A.21. The first of these becomes fifth order, and the second, sixth order when \( \frac{dY}{dT} \) is independent of \( Y \).

\[
\begin{array}{cccc|c}
0 & \frac{(4 - \sqrt{6})}{10} & \frac{(4 - \sqrt{6})}{10} & \text{King} \\
\frac{(4 + \sqrt{6})}{10} & -(\frac{11 + 4\sqrt{6}}{25}) & \frac{(42 + 13\sqrt{6})}{50} \\
1 & \frac{(1 + 5\sqrt{6})}{4} & -(\frac{3 + 2\sqrt{6}}{2}) & \frac{(9 - \sqrt{6})}{4} \\
\hline
0 & \frac{(16 - \sqrt{6})}{36} & \frac{(16 + \sqrt{6})}{36} & \frac{1}{9}
\end{array}
\]

\[
\begin{array}{c|cccc}
0 & \frac{(5 - \sqrt{5})}{10} & \frac{(5 - \sqrt{5})}{10} & \frac{(5 + \sqrt{5})}{10} & \frac{- (5 + 3\sqrt{5})}{20} \\
2/5 & 2/5 & \frac{- (5 + 3\sqrt{5})}{4} & \frac{(3 + \sqrt{5})}{4} & \frac{1}{4} \\
3/5 & -3/20 & \frac{1}{4} & \frac{5}{4} & \frac{1}{2} \\
1 & \frac{-1 + 5\sqrt{5}}{4} & \frac{- (5 + 3\sqrt{5})}{4} & \frac{(5 - \sqrt{5})}{2} & \frac{1}{12} \\
\end{array}
\]


\[
\begin{array}{c|cccc}
0 & \frac{1}{12} & \frac{5}{12} & \frac{5}{12} & \frac{1}{12} \\
2/5 & \frac{2}{5} & \frac{2}{5} & \frac{2}{5} & \frac{2}{5} \\
3/5 & \frac{-3/20}{3/4} & \frac{3/4}{3/4} & \frac{3/4}{3/4} & \frac{3/4}{3/4} \\
1 & \frac{19/44}{40/44} & \frac{-15/44}{40/44} & \frac{40/44}{40/44} & \frac{40/44}{40/44} \\
\end{array}
\]

\[
\begin{array}{c|cccc}
55/360 & 125/360 & 125/360 & 55/360 \\
\end{array}
\]

Hull and Johnston [13] point out that, \(c_2 \approx 0.35\) and \(c_3 \approx 0.45\) (solve deriving equations for remaining coefficients) lead to a minimum TEB. As previously mentioned in A.1-2, these bounds are usually larger than the actual error.

Lapidus and Seinfeld [6] list four methods; one of these, the Gill form, is optimized for round-off errors (has a larger TEB than Ralson's optimum).
Tests made by the author showed that the Gill and Classic forms are equal on accuracy while the Ralston form is superior to both these.
Hull [19] gives a good discussion on the problem of optimizing Runge-Kutta and predictor-corrector type methods.

The ambiguity of the minimum TEB measure can be pointed out; according to Luther and Sierra [14], using their own minimum TEB measure, the Kutta form, eq. A.27 is optimum for truncation error.

Blum [15] claims to wipe out the optimum round-off error advantage of the Gill form, eq. A.25, by modifying the arithmetic (computer) sequence of the Kutta form, eq. A.27, making it comparable to the Gill form. Fyfe [16] extends this procedure to any RKE(4,4) (though these modifications require extra programming effort).

Lawson [17] derives a RKE(4,4) with an extended region of stability; for integrating ordinary differential equations with large Lipschitz constants. This is comparable to the other RKE(4,4)'s on accuracy but can take larger step-sizes. Gates [26] gives a RKE(4,5).

Fehlberg [12] lists two methods with his usual feature of automatic step size control (see section A.1-2). Also eqs. A.16 and A.17, Fehlberg third order methods, have fourth order methods coupled, these can be used separately as fourth orders.

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1/9</th>
<th>2/9</th>
<th>1/3</th>
<th>1/12</th>
<th>1/4</th>
<th>3/4</th>
<th>69/128</th>
<th>-243/128</th>
<th>135/64</th>
</tr>
</thead>
<tbody>
<tr>
<td>p=4</td>
<td>1</td>
<td>-17/12</td>
<td>27/4</td>
<td>-27/5</td>
<td>16/15</td>
<td>------</td>
<td>------</td>
<td>--------</td>
<td>----------</td>
<td>--------</td>
</tr>
<tr>
<td>p=5</td>
<td>5/6</td>
<td>65/432</td>
<td>-5/16</td>
<td>13/16</td>
<td>4/27</td>
<td>5/144</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>9/20</td>
<td>16/45</td>
<td>1/12</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td>--------</td>
<td>----------</td>
<td>--------</td>
</tr>
<tr>
<td></td>
<td>47/450</td>
<td>12/25</td>
<td>32/225</td>
<td>1/30</td>
<td>6/25</td>
<td>5/144</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Fehlberg tested the above three methods, eqs. A.28, A.29 and A.30. Eq. A.28 was the best on accuracy, while eq. A.29 was the
worst. On time eq. A.29 was the best while eq. A.30 was the worst. All these three methods, by the same test were faster and more accurate than the Kutta form, eq. A.27. The time savings of these three methods is a result of their automatic step size control feature.

Sarafyan [37] shows that the Classic form, eq. A.26, has embedded within it first, second and third order methods. This is naturally expected of a fourth order method. This embedding principle of Sarafyan can be used to monitor step size. Using eq. A.26:

\[ Y_{n+1}^1 = Y_n + k_1, \quad Y_{n+1}^2 = Y_n + k_2 \text{ with } Y_{n+1}^4 \text{ available by the complete calculation of the method. So here } Y_{n+1}^2 \text{ (second order accuracy } 1/n) \]

is available at no extra cost. Comparing \( Y_{n+1}^2 \) and \( Y_{n+1}^4 \) gives step size control. \( Y_{n+1}^3 \) is not directly available, it is available at \( \frac{3}{4}h \) and not at \( h \); so to use it for step size control requires extra effort.

This embedding principle can be extended to any order method, and is a built in advantage of the Runge-Kutta methods.

Various authors have studied the errors involved in fourth order Runge-Kutta methods and developed embedding methods to check these errors and hence institute step size control; regions of stability have also been studied. These authors are: Merson [29], Scraton [20], England [21], Ceschino and Kuntzmann [32], Collatz [33], Lotkin [34], Chai [35], Sarafyan [36,37], Karim [38], Warten [29], Shampaine and Watts [40], Christiansen [41], and O'Regan [42].

Henrici [43] shows how to control round-off error.

Computer programs using fourth order processes with automatic step size control are listed in Basnett [69] (uses the England [31] method), and in Jones [70] (uses the Classic, eq. A.26, method).
The author, using the same principle outlined in section A.1-3, has developed two new classes of RKE(4,4)':s:

1. Back-step RKE(4,4):

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>-.1</th>
<th>.9</th>
<th>1</th>
<th>Optimum</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>A.31</td>
</tr>
<tr>
<td>-.1</td>
<td>-.1</td>
<td></td>
<td>.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.9</td>
<td>4.65</td>
<td>-3.75</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>7.44</td>
<td>-6.24</td>
<td>-.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.06</td>
<td>-.61</td>
<td>1.11</td>
<td>-.56</td>
<td></td>
</tr>
</tbody>
</table>

Error Bound (TEB) < .22254 ML^4. If for this method c_3 = .1 and c_2 = .9 was used, the bound would be < .50641 ML^4, more than twice that of eq. A.31. This happens in every case, for the coefficients of a back step method; so it is better to have c_2 as the negative back step coefficient rather than c_3. One function evaluation per step is saved here as the k_3 of (T_n, Y_n) can be used as the k_2 of (T_{n+1}, Y_{n+1}). This method is not self-starting, and needs a regular fourth order method with c_3 = .9 to start it.

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>-1/2</th>
<th>1/2</th>
<th>1</th>
<th>A.32</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>-1/2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-1/2</td>
<td></td>
<td>-1/2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/2</td>
<td></td>
<td>3/4</td>
<td>-1/4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>-2</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1/6</td>
<td>0</td>
<td>2/3</td>
<td>1/6</td>
<td></td>
</tr>
</tbody>
</table>

Error Bound (TEB) < .26181 ML^4. This method is close to the optimum, has simple coefficients, and can use either the Classic or the Gill form as a starting method.
II. Front Step Method $(4,3)$:

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>0.01</th>
<th>.01</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
</tr>
<tr>
<td>1.01</td>
<td>-50.52</td>
<td>51.53</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>-48.557981</td>
<td>49.548558</td>
<td>.009423</td>
</tr>
<tr>
<td>8.080858</td>
<td>8.585858</td>
<td>-8.085804</td>
<td>8.580804</td>
</tr>
</tbody>
</table>

Error Bound (TEB) < .23178 ML$^4$. Here again, as in the case of Back Step methods interchanging values of $c_2$ and $c_3$ results in a bound nearly twice as large, i.e., < .43719 ML$^4$ for $c_2 = 1.01$ and $c_3 = .01$. For this method the $k_3$ of $(T_n, Y_n)$ can be used as the $k_2$ of $(T_{n+1}, Y_{n+1})$. One major advantage over the Back Step methods is that the Front Step methods are self starting. The optimum Front Step eq. A.33 has a bound not much larger than the optimum Back Step eq. A.31; so the Front Step methods should have great potential.

As there are only two free parameters in the fourth order case, a back and front step $(V = 2)$ fourth order method is not possible; but as the fifth order case has five free parameters a back and front step fifth order is possible. The author is working on such fifth and higher order methods.

On comparing error bounds with the other fourth order methods it is seen that the bounds of the Front and Back step methods are much larger; this is to be expected, and is the price paid for achieving $V = 3$ in a fourth order method.

A.1-5 RKE$(5,6)$:

The RKE$(5,6)$'s are claimed by many exponents, in the Runge-Kutta field, to be the best compromise between accuracy and computational efficiency (computer time).
The deriving equations are presented in Butcher [3], Fehlberg [4], Luther and Konen [20], Luther [21], Konen and Luther [22]. These also give general solutions to the deriving equations. The last three references explore the complete range of solutions possible, in terms of two and one parameter families. These lead to various Newton-Cotes, Gauss, Radan and Lobatto family of formulae. Cassity [23], Cassity [24] and Lawson [25] also state and solve the deriving equations generally in terms of various free parameters.


Lapidus and Seinfeld [6] also list a number of RKE (5,6) including the Kutta form corrected by Nystom.

\[
\begin{array}{cccccccc}
 0 & & & & & & & \\
 1/3 & 1/3 & & & & & & \text{Nystrom} \\
 1 & 1/4 & -12/4 & 15/4 & & & & \\
 2/3 & 6/81 & 90/81 & -50/81 & 8/81 & & & \\
 4/5 & 6/75 & 36/75 & 10/75 & 8/75 & 0 & & \\
\end{array}
\]

\begin{array}{cccccccc}
 & 23/192 & 0 & 125/192 & 0 & -81/192 & 125/192 & \\
\end{array}

I. RKE(5,6) of Newton-Cotes Quadrative Family:

\[
\begin{array}{cccccccc}
 0 & & & & & & & \\
 1 & 1 & & & & & & \text{Luther [21]} \\
 1 & 1/2 & 1/2 & & & & & A.35 \\
 1/4 & 14/64 & 5/64 & -3/64 & & & & \\
 1/2 & -12/96 & -12/96 & 8/96 & 64/96 & & & \\
 3/4 & 0 & -9/64 & 5/64 & 16/64 & 36/64 & & \\
\end{array}
\]

\begin{array}{cccccccc}
 & 7/90 & 0 & 7/90 & 32/90 & 12/90 & 32/90 & \\
\end{array}
This Butcher form, eq. A.38, is recommended by Lapidus and Seinfeld [6] (after extensive tests) as the best RKE $(p,v)$ to use. Also Sarafyan [44] uses this form for an ingenious error analysis,
hence this form is highly recommended. The author's test show this
method to be the best RKE(5,6) on accuracy.

<table>
<thead>
<tr>
<th>0</th>
<th></th>
<th>Butcher [18]</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1/2</td>
<td>-1/2</td>
<td></td>
</tr>
<tr>
<td>1/4</td>
<td>5/16 -1/16</td>
<td>A.39</td>
</tr>
<tr>
<td>1/2</td>
<td>-3/4 1/4 1</td>
<td></td>
</tr>
<tr>
<td>3/4</td>
<td>3/16 0 0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0 -1/7 12/7 -12/7 8/7</td>
<td></td>
</tr>
<tr>
<td>7/90</td>
<td>0 32/90 12/90 32/90 7/90</td>
<td></td>
</tr>
</tbody>
</table>

This Butcher form is of the back step type, discussed in section
A.1-3. This allows in all steps after the first, to use $k_4$ from
$(T_n, Y_n)$ for the $k_2$ of $(T_{n+1}, Y_{n+1})$. Hence, overall, this method
requires $V = 5$ instead of $V = 6$. Another such method is (not of the
Newton-Cotes family):

<table>
<thead>
<tr>
<th>0</th>
<th></th>
<th>Butcher [18]</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1/5</td>
<td>-1/5</td>
<td></td>
</tr>
<tr>
<td>2/5</td>
<td>4/5 -2/5</td>
<td>A.40</td>
</tr>
<tr>
<td>1/3</td>
<td>7/36 0 5/36</td>
<td></td>
</tr>
<tr>
<td>4/5</td>
<td>0 0 4/5 0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1/4 0 -35/4 54/7 25/14</td>
<td></td>
</tr>
<tr>
<td>5/48</td>
<td>0 0 27/56 125/336 1/24</td>
<td></td>
</tr>
</tbody>
</table>

Here $c_2 = -1/5$ $c_5 = 4/5$ so allows the use of $k_5$ from $(T_n, Y_n)$ for
the $k_2$ of $(T_{n+1}, Y_{n+1})$. The larger number of zeros present in eq.
A.40 as compared to eq. A.39 should make eq. A.40 more computationally
efficient than eq. A.39.
The extra programming effort required to save the k's from a previous step, for use at a present step, is trivial.

Another Butcher [18] method, which can be used to start eq. A.40, is:

\[
\begin{array}{c|cccc}
0 & & & & \\
1/5 & 1/5 & & & \\
2/5 & 0 & 2/5 & & \\
1/3 & 7/36 & 0 & 5/36 & \\
4/5 & 0 & 0 & 4/5 & 0 \\
1 & 1/4 & 0 & -35/4 & 54/7 & 25/14 \\
\hline
5/48 & 0 & 0 & 27/56 & 125/336 & 1/24
\end{array}
\]

Butcher [18]

A.41

Lawson [25] claims a form, similar to his RKE(4,4). This method has an extended region of stability:

\[
\begin{array}{c|cccc}
0 & & & & \\
1/2 & 1/2 & & & \\
1/4 & 3/16 & 1/16 & & \\
1/2 & 0 & 0 & 1/2 & \\
3/4 & 0 & -3/16 & 6/16 & 9/16 \\
1 & 1/7 & 4/7 & 6/7 & -12/7 & 8/7 \\
\hline
7/90 & 0 & 32/90 & 12/90 & 32/90 & 7/90
\end{array}
\]

Lawson-form

A.42

The author's tests confirmed that this Lawson form, eq. A.42, does have a larger region of stability than other RKE(5,6)'s, and so can take larger step sizes. On accuracy this form is as good as the Bulcher form, eq. A.38. These two are the best RKE(5,6)'s on accuracy.
II. **RKE(5,6) of the Gauss Quadrature family:**

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>Luther and Konen [20]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1/2</td>
<td></td>
<td>3/8 1/8</td>
<td>A.43</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>-1/2 -1/2 4/2</td>
<td></td>
</tr>
<tr>
<td>(5 - √15)/10</td>
<td></td>
<td>-√15/100 10/100 (60 - 8√15)/100 √15/100</td>
<td></td>
</tr>
<tr>
<td>(5 + √15)/10</td>
<td></td>
<td>(-6 - √15)/20 2/20 12/20 (6 - √15)/20 4√15/20</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>8/18 5/18 5/18</td>
</tr>
</tbody>
</table>

III. **RKE(5,6) of the Radau Quadrature family:**

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>4/11 9/50 11/50 11/4 15/4 16</th>
<th>Luther and Konen [20]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>4/11</td>
<td></td>
</tr>
<tr>
<td>4/11</td>
<td></td>
<td>9/50 11/50</td>
<td>A.44</td>
</tr>
<tr>
<td>2/5</td>
<td></td>
<td>0 -11/4 15/4</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>(81 + 9√6)/600 0 (255 - 55√6)/600 (24 - 14√6)/600</td>
<td></td>
</tr>
<tr>
<td>(6 - √6)/10</td>
<td></td>
<td>(81 + 9√6)/600 0 (255 + 55√6)/600 (24 + 14√6)/600 0</td>
<td></td>
</tr>
<tr>
<td>(6 + √6)/10</td>
<td></td>
<td>4/36 0 0 0 (16 + √6)/36 (16 - √6)/36</td>
<td></td>
</tr>
</tbody>
</table>

The deriving equations for the Radau family of RKE(5,6)’s are given and solved in terms of free parameters in Luther [21] and Konen and Luther [22]. The above equation is one such solution.
IV. RKE(5,6) of the Lobatto Quadrative family:

<table>
<thead>
<tr>
<th></th>
<th>(5 - √5)</th>
<th>(5 + √5)</th>
<th>(5 - √5)</th>
<th>(5 + 3√5)</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1/2</td>
<td>2/10</td>
<td>6/10</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1/4</td>
<td>1/4</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Luther and Konen [20]

A.45

<table>
<thead>
<tr>
<th></th>
<th>(5 - √5)</th>
<th>(5 + √5)</th>
<th>(5 - √5)</th>
<th>(5 + 3√5)</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1/12</td>
<td>0</td>
<td>5/12</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Luther [21]

A.46

<table>
<thead>
<tr>
<th></th>
<th>(5 - √5)</th>
<th>(5 + √5)</th>
<th>(5 - √5)</th>
<th>(5 + 3√5)</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1/12</td>
<td>0</td>
<td>5/12</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Konen and Luther [22] solve the deriving equations of the Lobatto family in terms of free parameters. The above two equations are two such solutions.

The advantage of the Newton-Cotes, Gauss, Radau and Lobatto families is that when \( \frac{dy}{dT} \) is independent of \( Y \), they become sixth order quadrative formulae of these families.
Luther and Konen [20] point out that, on basis of computational efficiency, due to the number of zeros present, the Gauss and Radau forms are better than the Lobatto forms, which in turn is better than the Newton Cotes form. While a round-off error minimization; following the principle outlined in Gates [26], shows that on accuracy (based on round-off error), the Lobatto form is better than the other three.

Shanks [27] developed a class of RKE(5,5)'s rather than RKE(5,6)'s. This was done by solving the non-linear deriving equation approximately, not exactly, resulting in one less function evaluation. The theory for reducing V for RKE's is also given here for p ≤ 7, making it possible to derive more efficient formulae as compared to the conventional approach. However, it must be pointed out that (in the opinion of the author) methods derived this way are not exactly p order, though they are greater than p-1 order. This is because the truncation error, in such methods has a larger bound. The author's tests confirmed that Shank's RKE(5,5)'s are faster, though less accurate than, RKE(5,6)'s.

<table>
<thead>
<tr>
<th>0</th>
<th>( \frac{1}{1000N} )</th>
<th>( \frac{1}{1000N} )</th>
<th>( \frac{1}{1000N} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{3}{10} )</td>
<td>( \frac{-450N + 3}{10} )</td>
<td>( \frac{450N}{10} )</td>
<td>( \frac{A.47}{1000N} )</td>
</tr>
<tr>
<td>( \frac{3}{4} )</td>
<td>( \frac{2250N - 9}{8} )</td>
<td>( \frac{2250N}{8} )</td>
<td>( \frac{8}{1} )</td>
</tr>
<tr>
<td>1</td>
<td>( \frac{-103500N + 459}{81} )</td>
<td>( \frac{103500N}{81} )</td>
<td>( \frac{490}{1} )</td>
</tr>
</tbody>
</table>

\[ \frac{105}{1134} \quad 0 \quad \frac{500}{1134} \quad \frac{448}{1134} \quad \frac{81}{1134} \]

Shanks [27]
Shanks suggests that $1200 \, Nh^2 > 1$ be chosen to give a valid fifth order method. The author tested values of $N$ from 100,000 to .0001 and found that $N \geq 10$ gave discrepancies in the coefficients, while $N < 10$ does give a valid method. Shanks suggests $N = 9$.

The author's test with various values of $N$ showed that $N = 5$ gave the most accurate method.

The author modified the Shanks form to give a method of four instead of five stages. This was done by using $N = 100$, resulting in $c_2 = \frac{1}{100,000}$. As $c_2$ is much smaller than $c_3$, $c_4$ or $c_5$, $c_2 = 0$ was used. This makes $k_2 = k_1$ and hence one function evaluation is saved.

\[
\begin{array}{c|ccccc}
0 & & & & & \\
3 & 0 & & & & \\
5 & \frac{-450N + 3}{10} & \frac{450N}{10} & & & \\
\frac{3}{4} & \frac{2250N - 9}{8} & \frac{-2250N}{8} & \frac{15}{9} & & \\
1 & \frac{-103500N + 459}{81} & \frac{103500N}{81} & \frac{-490}{81} & \frac{112}{81} & \\
\hline
& \frac{105}{1134} & 0 & \frac{500}{1134} & \frac{448}{1134} & \frac{81}{1134} \\
\end{array}
\]

Modified Shanks

A.48

The author's tests showed that this method is faster though less accurate than the other Shanks forms.

Sarafyan [28] develops similar type methods with step size control, based on the embedding principle outlined in section A.1-4. But Sarafyan prefers to control his p order methods by embedded p-i, i = 1, 2, 3---, order methods. This requires no extra function evaluations, so is faster than Fehlberg's methods. But as Fehlberg uses a p order method controlled by a p+l order method, his error estimate will be better than Sarafyan; and hence a larger step size is possible with Fehlberg's methods, at the cost of extra function evaluations. However the Sarafyan methods can obviously be used in the same way as the Fehlberg methods, by using the p order method to control a p-1 order solution.
The Sarafyan and Fehlberg step size control principles are convenient for use and are extendible to any order method.

A Sarafyan fifth order with an embedded fourth order, is eq. A.30 listed in sections A.1-4. Sarafyan [28] gives the deriving equations of such methods and solves them generally, giving six such embedded methods.

<table>
<thead>
<tr>
<th>p=2</th>
<th>( \frac{1}{2} )</th>
<th>( \frac{1}{2} )</th>
<th>( \frac{1}{4} )</th>
<th>( \frac{1}{4} )</th>
<th>Sarafyan</th>
</tr>
</thead>
<tbody>
<tr>
<td>p=4</td>
<td>( \frac{1}{6} )</td>
<td>( 0 )</td>
<td>( -1 )</td>
<td>( 2 )</td>
<td>( \frac{24}{1000} (11 \ 15 \ -2 \ 1) )</td>
</tr>
<tr>
<td>p=5</td>
<td>( \frac{3}{256} )</td>
<td>( (18 \ 24 \ 40 \ 7 \ -25) )</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| p=2 | \( \frac{1}{2} \) | \( 0 \) | \( 1 \) | \( \frac{1}{6} \) | \( (1 \ 0 \ 2 \ 1) \) | \( \frac{1}{54} \) | \( (7 \ 0 \ 108 \ 0 \ -125 \ 64) \) |
|-----|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------|

| p=2 | \( \frac{1}{2} \) | \( \frac{1}{2} \) | Same as | eq. A.50 | | | | |
|-----|-----------------|-----------------|----------|-------------| | | | |
| p=4 | \( \frac{8}{10} \) | \( \frac{16}{1000} (13 \ 10 \ 24 \ 3) \) | | | | | |
| p=5 | \( \frac{7}{10} \) | \( \frac{7}{80} (3 \ 4 \ 0 \ 0 \ 1) \) | | | | Same as | eq. A.50 | p=2 |
|-----|-----------------|-----------------|----------|-------------| | | | |-----------------|-----------------|-----------|-----------|
| p=4 | \( \frac{1}{504} \) | \( (69 \ 0 \ 616 \ -56 \ 875 \ -1000) \) | | | | | | | | | | | |
| p=2 | 1/2 | \( \frac{1}{2} \) | \begin{array}{c}
\text{Same as} \\
\text{eq. A.50}
\end{array} |
|-----|-----|----------------|----------------|
| p=4 | 1/3 | \( \frac{1}{27} \) | \begin{pmatrix}
7 & 10 & 0 & 1
\end{pmatrix} |
| p=5 | 3/5 | \( \frac{3}{8} \) | \begin{pmatrix}
0 & -30 & 0 & -12 & 45
\end{pmatrix} |

| p=2 | \( \frac{1}{2} \) | \begin{array}{c}
\text{Same as} \\
\text{eq. A.50}
\end{array} |
|-----|-----|----------------|----------------|
| p=4 | 1/3 | \( \frac{1}{27} \) | \begin{pmatrix}
7 & 10 & 0 & 1
\end{pmatrix} |
| p=5 | 2/10 | \( \frac{16}{1000} \) | \begin{pmatrix}
28 & -125 & 546 & 54 & -378
\end{pmatrix} |

| p=2 | \( \frac{1}{2} \) | \begin{array}{c}
\text{Same as} \\
\text{eq. A.50}
\end{array} |
|-----|-----|----------------|----------------|
| p=4 | 1/3 | \( \frac{1}{27} \) | \begin{pmatrix}
7 & 10 & 0 & 1
\end{pmatrix} |
| p=5 | 2/10 | \( \frac{16}{336} \) | \begin{pmatrix}
14 & 0 & 0 & 35 & 162 & 125
\end{pmatrix} |
<table>
<thead>
<tr>
<th>p</th>
<th>( \pm 2 )</th>
<th>( \pm 4 )</th>
<th>( \pm 5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Same as ( \text{eq. A.50} )</td>
<td>Sarafyan A.54</td>
<td></td>
</tr>
<tr>
<td>( \pm 2 )</td>
<td>( \frac{1}{2} )</td>
<td>( \frac{1}{2} )</td>
<td>( \frac{1}{2} )</td>
</tr>
<tr>
<td>( \pm 4 )</td>
<td>( \frac{1}{2} )</td>
<td>( \frac{1}{2} )</td>
<td>( \frac{1}{2} )</td>
</tr>
<tr>
<td>( \pm 5 )</td>
<td>( \frac{1}{2} )</td>
<td>( \frac{1}{2} )</td>
<td>( \frac{1}{2} )</td>
</tr>
</tbody>
</table>

\[
\begin{array}{cccc}
\frac{2}{3} & \frac{1}{27} & 7 & 10
\end{array}
\]

\[
\begin{array}{cccc}
.2478 & .035 & .0896 & .504
\end{array}
\]

or

\[
\begin{array}{cccc}
.014(177 & 250 & 64 & 36 & -27)
\end{array}
\]

Same as \( \pm 2 \) eq. A.50

\[
\begin{array}{cccc}
\frac{1}{84} & (11 & 0 & 140 & -567 & 500 & 0)
\end{array}
\]

\[
\begin{array}{cccc}
\frac{1}{84} & (11 & 0 & 140 & -567 & 500 & 0)
\end{array}
\]

\[
\begin{array}{cccc}
\frac{1}{84} & (11 & 0 & 140 & -567 & 500 & 0)
\end{array}
\]

Same as Sarafyan A.55

\[
\begin{array}{cccc}
.1(6 - \sqrt{6}) & .002 & (93 + 2\sqrt{6}) & 0
\end{array}
\]

\[
\begin{array}{cccc}
4(56 - 11\sqrt{6}) & (3 - 8\sqrt{6})
\end{array}
\]

<table>
<thead>
<tr>
<th>p</th>
<th>( \pm 2 )</th>
<th>( \pm 4 )</th>
<th>( \pm 5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \pm 2 )</td>
<td>( \frac{1}{2} )</td>
<td>( \frac{1}{2} )</td>
<td>( \frac{1}{2} )</td>
</tr>
<tr>
<td>( \pm 4 )</td>
<td>( \frac{1}{2} )</td>
<td>( \frac{1}{2} )</td>
<td>( \frac{1}{2} )</td>
</tr>
<tr>
<td>( \pm 5 )</td>
<td>( \frac{1}{2} )</td>
<td>( \frac{1}{2} )</td>
<td>( \frac{1}{2} )</td>
</tr>
</tbody>
</table>

\[
\begin{array}{cccc}
\frac{1}{36} & 4 & 0 & 0
\end{array}
\]

\[
\begin{array}{cccc}
(16 + \sqrt{6}) & (16 - \sqrt{6})
\end{array}
\]

<table>
<thead>
<tr>
<th>p</th>
<th>( \pm 2 )</th>
<th>( \pm 4 )</th>
<th>( \pm 5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \pm 2 )</td>
<td>( \frac{1}{2} )</td>
<td>( \frac{1}{2} )</td>
<td>( \frac{1}{2} )</td>
</tr>
<tr>
<td>( \pm 4 )</td>
<td>( \frac{1}{2} )</td>
<td>( \frac{1}{2} )</td>
<td>( \frac{1}{2} )</td>
</tr>
<tr>
<td>( \pm 5 )</td>
<td>( \frac{1}{2} )</td>
<td>( \frac{1}{2} )</td>
<td>( \frac{1}{2} )</td>
</tr>
</tbody>
</table>

\[
\begin{array}{cccc}
\frac{1}{36} & 4 & 0 & 0
\end{array}
\]

\[
\begin{array}{cccc}
(16 + \sqrt{6}) & (16 - \sqrt{6})
\end{array}
\]
All these embedding forms eq. A.50 to eq. A.55 are comparable to other conventional fifth order methods on accuracy but their step size control makes them faster.

Sarafyan [44] employs an ingenious step size control with:

\[
\begin{array}{cccccc}
0 & 1/2 & 1/2 & \text{Sarafyan} \\
1/2 & 1/4 & 1/4 & A.56 \\
1 & 0 & -1 & 2 \\
3/2 & 3/8 & 0 & 0 & 9/8 \\
2 & -6/7 & 4/7 & 24/7 & -24/7 & 16/7 \\
7/45 & 0 & 32/45 & 12/45 & 32/45 & 7/45 \\
\end{array}
\]

This form is gotten from the Butcher form, eq. A.38, by multiplying all coefficients by a factor of 2. Sarafyan shows that in going from \( (T_n, Y_n) \) to \( (T_{n+1}, Y_{n+1}) \); using eq. A.38 with \( h \), \( Y_{n+1} \) is available to a fourth order accuracy; at the same time, using eq. A.50 with \( 2h \), \( Y_{n+2} \) is available to a fifth order accuracy. Net result - overall \( V=3 \) is achieved although the RKE(5,6) of Butcher is used; and by comparing fourth and fifth order \( Y_{n+1} \)'s step size can be controlled.

In the opinion of the author, this Sarafyan method should be more accurate than eq. A.50 to eq. A.55, because the most accurate fifth order, eq. A.38, is used here.

A.1-6 RKE(6,7):

The deriving equations are presented in Butcher [3], Fehlberg [4], and Rosen [5]. The first two also give the general solution of these equations, in terms of various free parameters.
In Butcher [3] are listed four RKE(6,7)'s

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th>Butcher Newton-Cotes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>A.57</td>
</tr>
<tr>
<td>1/3</td>
<td>1/3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2/3</td>
<td>0</td>
<td>2/3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/3</td>
<td>1/12</td>
<td>1/3</td>
<td>-1/12</td>
<td></td>
</tr>
<tr>
<td>1/2</td>
<td>-1/16</td>
<td>9/8</td>
<td>-3/16</td>
<td>-3/8</td>
</tr>
<tr>
<td>1/2</td>
<td>0</td>
<td>9/8</td>
<td>-3/8</td>
<td>-3/4</td>
</tr>
<tr>
<td>1</td>
<td>9/44</td>
<td>-9/11</td>
<td>63/64</td>
<td>18/11</td>
</tr>
<tr>
<td>11/120</td>
<td>0</td>
<td>27/40</td>
<td>27/40</td>
<td>-4/15</td>
</tr>
</tbody>
</table>

A back step and front step type method (see sections A.1-3 and A.1-4) is also listed.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th>Butcher Newton-Cotes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>A.58</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2/3</td>
<td>4/9</td>
<td>2/9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/3</td>
<td>11/36</td>
<td>1/9</td>
<td>-1/12</td>
<td></td>
</tr>
<tr>
<td>-1/3</td>
<td>151/36</td>
<td>29/9</td>
<td>-7/4</td>
<td>-6</td>
</tr>
<tr>
<td>4/3</td>
<td>-112/9</td>
<td>-116/9</td>
<td>32/3</td>
<td>18</td>
</tr>
<tr>
<td>1</td>
<td>-5/4</td>
<td>-29/9</td>
<td>397/276</td>
<td>152/69</td>
</tr>
</tbody>
</table>

In eq. A. 58 $k_3$ from $(T_n, Y_n)$ can be used for the $k_5$ of $(T_{n+1}, Y_{n+1})$; and the $k_6$ of $(T_n, Y_n)$ can be used as the $k_4$ of $(T_{n+1}, Y_{n+1})$. So overall $V=4$ is achieved, hence this method is highly recommended.
<table>
<thead>
<tr>
<th></th>
<th>Butcher Newton-Cotes</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/2</td>
<td>1/2</td>
<td></td>
</tr>
<tr>
<td>2/3</td>
<td>2/9 4/9</td>
<td></td>
</tr>
<tr>
<td>1/3</td>
<td>7/36 2/9 -1/12</td>
<td></td>
</tr>
<tr>
<td>5/6</td>
<td>-35/144 -55/36 35/48 15/8</td>
<td></td>
</tr>
<tr>
<td>1/6</td>
<td>-1/360 -11/36 -1/8 1/2 1/10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>13/200 0 11/40 11/40 4 4 13 200</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Butcher Lobatto</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 \sqrt{5}/10</td>
<td>5 \sqrt{5}/10 5 \sqrt{5}/10</td>
<td></td>
</tr>
<tr>
<td>5 \sqrt{5}/10</td>
<td>\sqrt{5}/10 5 \sqrt{5}/10 2\sqrt{5}/10</td>
<td></td>
</tr>
<tr>
<td>5 \sqrt{5}/10</td>
<td>-15/10 -5\sqrt{5}/10 -1/10 15 \sqrt{5}/10</td>
<td></td>
</tr>
<tr>
<td>5 \sqrt{5}/10</td>
<td>5 \sqrt{5}/60 0 1/6 15 \sqrt{5}/60</td>
<td></td>
</tr>
<tr>
<td>5 \sqrt{5}/10</td>
<td>5 \sqrt{5}/60 0 9 \sqrt{5}/12 1/6 -3 \sqrt{3}/5</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1/6 0 -55/12 -25 \sqrt{5}/12 -25 \sqrt{5}/12 5 \sqrt{2} 5 + \sqrt{5}/2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1/12 0 0 0 5/12 5/12 1/12</td>
<td></td>
</tr>
</tbody>
</table>
Luther [45] derives an 'optimum' Lobatto form:

\[
\begin{array}{cccc}
0 & 1 & & \\
1 & & \frac{1}{6} & [ \ 3 \ 1 \ ] \\
\frac{1}{2} & & \frac{1}{27} & [ \ 8 \ 2 \ 8 \ ] \\
\frac{3}{2} & & \frac{7}{32} & [ \ 3(3t - 7) - 8(7 - 2t) + 48(t - 2t) - 3(2t - 2t) ] \\
1 & & \frac{1}{180} & [ \ 15(2t + 7) - 60(t + 2t) + 120(t + 2t) - 120(t + 2t) - 24(49 + 21t) - 24(7 - 2t) ] \\
\end{array}
\]

All these quadrative forms eqs. A.57 to eq. A.61 become seventh order when \( \frac{dY}{dT} \) is independent of \( Y \).

Lawson [46] lists a form with an extended order of stability (similar analysis to his RKE(4,4) and RKE(5,6), but admits that its accuracy improvement is marginal compared to other sixth order formulae, in fact worse for some non-linear problems.

Fehlberg [4] lists a RKE\([6,8];(7,10)\], with the usual advantage of Fehlberg methods (see section A.1-2), namely automatic step size control and minimum TEB. Sarafyan [47] lists four RK(6,8) with their embedding advantages (similar to his fifth order methods, see section A.1-5). Huta's RKE(6,8) is included in this reference. Some, rather obscure, advantages are claimed for these so-called improved sixth order methods. One definite advantage is that all four are of the Newton-Cotes family and when \( \frac{dY}{dT} \) is independent of \( Y \) they become eighth order in accuracy.
\[ Y_6(T_n + h) = Y_n + \frac{1}{840} \left[ 41(k_1 + k_{18}) + 216(k_3 + k_7) + 27(k_4 + k_{15}) + 272k_5 \right] \]

where

\[
\begin{align*}
k_1 &= hf(T_n, Y_n) \\
k_2 &= hf(T_n + \frac{1}{9}h, Y_n + \frac{1}{9}k_1) \\
k_3 &= hf(T_n + \frac{1}{6}h, Y_n + \frac{1}{6}k_2 - 3k_3) \\
k_4 &= hf(T_n + \frac{1}{3}h, Y_n + \frac{1}{3} (k_1 - 3k_2 + 4k_3)) \\
k_5 &= hf(T_n + \frac{1}{2}h, Y_n + \frac{1}{2} (-5k_1 + 27k_2 - 24k_3 + 6k_4)) \\
k_6 &= hf(T_n + \frac{2}{3}h, Y_n + \frac{1}{9} (221k_1 - 981k_2 + 867k_3 - 102k_4 + k_5)) \\
k_7 &= hf(T_n + \frac{5}{6}h, Y_n + \frac{1}{48} (-183k_1 + 678k_2 - 472k_3 - 66k_4 + 80k_5 + 3k_6)) \\
k_8 &= hf(T_n + h, Y_n + \frac{1}{82} (716k_1 - 2079k_2 + 1002k_3 + 834k_4 - 454k_5 - 9k_6 + 72k_7))
\end{align*}
\]
\[ Y_6(T_n + h) = Y_n + \frac{1}{840} \left[ 41(k_1 + k_8) + 216(k_3 + k_7) + 27(k_4 + k_6) + 272k_5 \right] \]

where

\[ k_1 = hf(T_n, Y_n) \]

\[ k_2 = hf(T_n + \frac{1}{9}h, Y_n + \frac{1}{9}k_1) \quad \text{Sarafyan} \]

\[ k_3 = hf(T_n + \frac{1}{6}h, Y_n + \frac{1}{24}(k_1 + 3k_2)) \quad \text{A.64} \]

\[ k_4 = hf(T_n + \frac{1}{3}h, Y_n + \frac{1}{6}(k_1 - 3k_2 + 4k_3)) \]

\[ k_5 = hf(T_n + \frac{1}{2}h, Y_n + \frac{1}{8}(k_1 + 3k_4)) \]

\[ k_6 = hf(T_n + \frac{2}{3}h, Y_n + \frac{1}{3}(-4k_1 - 21k_2 + 46k_3 - 29k_4 + 10k_5)) \]

\[ k_7 = hf(T_n + \frac{5}{6}h, Y_n + \frac{1}{72}(-8k_1 + 99k_2 - 84k_3 + 44k_4 + 9k_6)) \]

\[ k_8 = hf(T_n + h, Y_n + \frac{1}{32}(107k_1 - 243k_2 + 354k_4 - 172k_5 - 36k_6 + 72k_7)) \]
\[ k_1 = hf(T_n, Y_n) \]
\[ k_2 = hf(T_n + \frac{1}{9}h, Y_n + \frac{1}{9}k_1) \] (Sarafyan)
\[ k_3 = hf(T_n + \frac{1}{6}h, Y_n + \frac{1}{24} (k_1 + 3k_2)) \] (A.66)
\[ k_4 = hf(T_n + \frac{1}{3}h, Y_n + \frac{1}{6} (k_1 - 3k_2 + 4k_3)) \]
\[ k_5 = hf(T_n + \frac{1}{2}h, Y_n + \frac{1}{8} (k_1 + 3k_4)) \]
\[ k_6 = hf(T_n + \frac{2}{3}h, Y_n + \frac{1}{9} (17k_1 - 63k_2 + 51k_3 + k_5)) \]
\[ k_7 = hf(T_n + \frac{5}{6}h, Y_n + \frac{1}{24} (-22k_1 + 33k_2 + 30k_3 - 58k_4 + 34k_5 + 3k_6)) \]
\[ k_8 = hf(T_n + h, Y_n + \frac{1}{82} (281k_1 - 243k_2 - 522k_3 + 876k_4 - 346k_5 - 36k_6 + 72k_7)) \]

Shanks [27] developed an almost (6,6) method on the same lines as his RKE(5,5), eq. A.47.

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>\frac{1}{300}</td>
<td>\frac{1}{300}</td>
</tr>
<tr>
<td>\frac{1}{5}</td>
<td>-\frac{29}{5}</td>
</tr>
<tr>
<td>\frac{3}{5}</td>
<td>\frac{323}{5}</td>
</tr>
<tr>
<td>\frac{14}{5}</td>
<td>\frac{510104}{810}</td>
</tr>
<tr>
<td>1</td>
<td>-\frac{417923}{77}</td>
</tr>
<tr>
<td>\frac{198}{3696}</td>
<td>0</td>
</tr>
</tbody>
</table>
A.1-7 RKE(7,10):

Butcher [3], Fehlberg [4], and Rosen [5] list the deriving equations.

Sarafyan [48] gives four such formulae, of his usual embedded types (see section A.1-4), with built in step size control. His best form on accuracy is:

\[
Y(T + h) = Y_n + k_1
\]

\[
Y_2(T + h) = Y_n + \frac{1}{8}(-55k_1 + 63k_2)
\]

\[
Y_7(T + h) = Y_n + \frac{1}{17280}[751(k_1 + k_{10}) + 3577(k_4 + k_9) + 1323(k_5 + k_8) + 2989(k_6 + k_7)]
\]

\[
k_1 = hf(T_n, Y_n)
\]

\[
k_2 = hf(T_n + \frac{4}{63}h, Y_n + \frac{4}{63}k_1)
\]

Sarafyan

\[
k_3 = hf(T_n + \frac{2}{21}h, Y_n + \frac{1}{42}(k_1 + 3k_2))
\]

A.68

\[
k_4 = hf(T_n + \frac{1}{7}h, Y_n + \frac{1}{28}(k_1 + 3k_3))
\]

\[
k_5 = hf(T_n + \frac{2}{7}h, Y_n + \frac{1}{7}(k_1 - 3k_3 + 4k_4))
\]

\[
k_6 = hf(T_n + \frac{3}{7}h, Y_n + \frac{3}{28}(k_1 + 3k_5))
\]

\[
k_7 = hf(T_n + \frac{4}{7}h, Y_n + \frac{2}{1827}(-101k_1 + 651k_4 - 477k_5 + 449k_6))
\]

\[
k_8 = hf(T_n + \frac{5}{7}h, Y_n + \frac{5}{65772}(-1881k_1 - 783k_3 + 10352k_4 - 3414k_5 + 5122k_7))
\]

\[
k_9 = hf(T_n + \frac{6}{7}h, Y_n + \frac{1}{2222850}(683663k_1 + 430650k_3 - 2032615k_4 + 2208930k_5 + 385270k_6 - 740735k_7 + 970137k_8))
\]

\[
k_{10} = hf(T_n + h, Y_n + \frac{1}{19601100}(-12175421k_1 - 11236050k_3 + 62891430k_4 - 43488585k_5 + 9947140k_6 + 51099720k_7 - 30879954k_8 + 13337100k_9)).
\]
\[ Y_1(T_n + h) = Y_n + k_1 \]

\[ Y_2(T_n + h) = Y_n + \frac{1}{2}(-k_1 + 3k_2) \]

\[ Y_4(T_n + h) = Y_n + \frac{1}{6}(k_1 + 4k_3 + k_4) \]

\[ Y_7(T_n + h) = Y_n + \frac{1}{840}[41(k_1 + k_{10}) + 216(k_5 + k_9) + 27(k_6 + k_8) + 272k_7] \]

\[ k_1 = hf(T_n, Y_n) \]

\[ k_2 = hf(T_n + \frac{1}{3}h, Y_n + \frac{1}{3}k_1) \]

\[ k_3 = hf(T_n + \frac{1}{2}h, Y_n + \frac{1}{8}[k_1 + 3k_2]) \]

\[ k_4 = hf(T_n + h, Y_n + \frac{1}{2}[k_1 - 3k_2 + 4k_3]) \]

\[ k_5 = hf(T_n + \frac{1}{6}h, Y_n + \frac{1}{648}[83k_1 + 32k_3 - 7k_4]) \]

\[ k_6 = hf(T_n + \frac{1}{3}h, Y_n + \frac{1}{54}[-3k_1 - 4k_3 + k_4 + 24k_5]) \]

\[ k_7 = hf(T_n + \frac{1}{2}h, Y_n + \frac{1}{5088}[-290k_1 - 524k_3 + 145k_4 + 1908k_5 + 1305k_6]) \]

\[ k_8 = hf(T_n + \frac{2}{3}h, Y_n + \frac{1}{1431}[292k_1 + 108k_3 + 13k_4 - 318k_5 + 753k_6 + 106k_7]) \]

\[ k_9 = hf(T_n + \frac{5}{6}h, Y_n + \frac{1}{68688}[14042k_1 + 11012k_3 - 4477k_4 + 5724k_5 - 6903k_6 + 6360k_7 + 31482k_8]) \]

\[ k_{10} = hf(T_n + h, Y_n + \frac{1}{4346}[-2049k_1 - 1836k_3 + 839k_4 + 5724k_5 - 4692k_6 + 12084k_7 - 9540k_8 + 3816k_9]) \].
\[
\begin{align*}
Y_1(T_n+h) &= Y_n + Y_n + \frac{1}{2}(-25k_1+27k_2) \\
Y_2(T_n+h) &= Y_n + \frac{1}{2}(-25k_1+27k_2) \\
Y_3(T_n+h) &= Y_n + \frac{1}{800}(141k_1+100k_2) + 216(k_5+k_6) + 27(k_6+k_8) + 272k_2 \\
Y_4(T_n+h) &= Y_n + \frac{1}{2}(-k_1+3k_2) \\
Y_5(T_n+h) &= Y_n + \frac{1}{34}(k_1+4k_3+k_4)
\end{align*}
\]

where
\[
\begin{align*}
k_1 &= hf(T_n,Y_n) \\
k_2 &= hf(T_n + \frac{1}{2}h, Y_n + \frac{1}{2}h_1) \\
k_3 &= hf(T_n + \frac{1}{2}h, Y_n + \frac{1}{2}(-k_1+3k_2)) \\
k_4 &= hf(T_n + \frac{1}{9}h, Y_n + \frac{1}{18}(-k_1+3k_2)) \\
k_5 &= hf(T_n + \frac{1}{9}h, Y_n + \frac{1}{18}(-k_1+3k_2)) \\
k_6 &= hf(T_n + \frac{1}{6}h, Y_n + \frac{1}{2}(-k_1+3k_2)) \\
S &= hf(T_n + \frac{1}{6}h, Y_n + \frac{1}{2}(-k_1+3k_2))
\end{align*}
\]
\[
Y_1(T_n + h) = Y_n + \frac{\beta}{\alpha}
\]
\[
Y_2(T_n + h) = Y_n - 8k + 9k_2
\]
\[
Y_3(T_n + h) = Y_n + \left\{ 41(k_1 + k_{10}) + 216(k_3 + k_5) + 27(k_6 + k_8) + 272k_7 \right\}
\]
\[
Y_4(T_n + h) = Y_n + \frac{\beta}{\alpha}k_1
\]
\[
Y_3(T_n + h) = Y_n + \frac{\beta}{2\alpha}(-k_1 + 3k_2)
\]
\[
Y_4(T_n + h) = Y_n + \frac{\beta}{3\alpha}(k_1 + 4k_3 + k_4)
\]
where

\[
k_1 = hf(T_n, Y_n)
\]
\[
k_2 = hf(T_n + \frac{1}{18}h, Y_n + \frac{1}{18}k_1)
\]
\[
k_3 = hf(T_n + \frac{1}{12}h, Y_n + \frac{1}{48}(k_1 + 3k_2))
\]
\[
k_4 = hf(T_n + \frac{1}{6}h, Y_n + \frac{1}{12}(k_1 - 3k_2 + 4k_3))
\]
\[
k_5 = hf(T_n + \frac{1}{6}h, Y_n + \frac{1}{36}(k_1 + 4k_3 + k_4))
\]
and

\[
k_6 = hf(T_n + \frac{1}{3}h, Y_n + \frac{1}{27}(6k_1 - 12k_3 + 221k_4 - 206k_5))
\]
\[
k_7 = hf(T_n + \frac{1}{2}h, Y_n + \frac{1}{42}(115k_1 + 448k_3 - 336k_4 + 215k_5))
\]
\[
k_8 = hf(T_n + \frac{2}{3}h, Y_n + \frac{1}{477}(17k_1 + 456k_3 - 839k_4 + 8268k_5 - 49k_6 + 212k_7))
\]
\[
k_9 = hf(T_n + \frac{5}{6}h, Y_n + \frac{1}{11448}(17367k_1 - 52248k_3 + 36218k_4 + 13144k_5 + 10188k_6 + 5247k_8))
\]
\[
k_{10} = hf(T_n + h, Y_n + \frac{1}{2175}(11407k_1 + 35012k_3 - 17490k_4 + 3979k_5 + 6572k_7 - 4770k_8 + 1908k_9))
\]
or

\[
k_6 = hf(T_n + \frac{1}{3}h, Y_n + \frac{1}{9}(2k_1 - 4k_3 + 109k_4 - 104k_5))
\]
\[
k_7 = hf(T_n + \frac{1}{2}h, Y_n + \frac{1}{42}(115k_1 + 448k_3 - 548k_4 + 212k_5 + 215k_6))
\]
\[
k_8 = hf(T_n + \frac{2}{3}h, Y_n + \frac{1}{477}(17k_1 + 456k_3 - 12426k_5 - 49k_6 + 212k_7))
\]
\[
k_9 = hf(T_n + \frac{5}{6}h, Y_n + \frac{1}{3816}(5789k_1 - 17416k_3 + 19846k_4 - 3392k_5 - 3396k_6 + 1749k_8))
\]
\[
k_{10} = hf(T_n + h, Y_n + \frac{1}{2175}(11407k_1 + 35012k_3 - 21277k_4 - 7844k_5 + 3979k_6 + 6572k_7 - 4770k_8 + 1908k_9)).
\]
Fehlberg [4] lists a RKE[(7,11);(8,13)]; with the usual advantages of this form (discussed in section A.1-2): automatic step size control and a minimized TEB.

\[
\begin{array}{cccccc}
0 & 27 & 1 & 36 & 1 & 12 \\
2 & 27 & 1 & 24 & 0 & 1 & 8 \\
5 & 12 & 5 & 12 & 0 & -25 & 16 & 25 & 16 \\
1 & 2 & 1 & 20 & 0 & 0 & 1 & 4 & 1 & 5 \\
5 & 6 & -25 & 108 & 0 & 0 & 125 & 108 & -65 & 27 & 125 & 54 \\
1 & 6 & 31 & 300 & 0 & 0 & 0 & 61 & 225 & -2 & 9 & 13 & 900 \\
2 & 3 & 2 & 0 & 0 & -33 & 6 & 704 & 45 & -107 & 67 & 9 & 90 & 3 \\
1 & 3 & -9 & 108 & 0 & 0 & 23 & 108 & 976 & 135 & 311 & 19 & 17 & 6 & 12 \\
\end{array}
\]

\(p=7\) \(1\) \(2383/4100\) \(0\) \(0\) \(-341/164\) \(4496/1025\) \(-301/82\) \(2133/4100\) \(45/82\) \(45/164\) \(18/41\)

\(p=8\) \(1\) \(-1777/4100\) \(0\) \(0\) \(-341/164\) \(4496/1025\) \(-289/82\) \(2193/4100\) \(51/82\) \(33/164\) \(12/41\) \(0\) \(1\)

Shanks [27] following the approach discussed in section A.1-5, lists a RKE(7,7) and a RKE(7,9). On a test system, as expected, the (7,9) was more accurate, while the (7,7) was faster.
\[ \begin{array}{l|rrr|rrr} 
\text{0} & 1 & 1 & 1 & 1 & 1 & 1 \\
\frac{1}{192} & \frac{1}{192} & \frac{1}{192} & \frac{1}{192} & \frac{1}{192} & \frac{1}{192} & \frac{1}{192} \\
\frac{1}{6} & \frac{1}{6} & (\ -15 & 16\ ) & \frac{1}{6} & (\ -15 & 16\ ) & \text{Shanks} \\
\frac{1}{2} & \frac{1}{186} & (\ 4867 & -5072 & 298\ ) & \frac{1}{186} & (\ 4867 & -5072 & 298\ ) & \text{A.73} \\
1 & \frac{1}{31} & (\ -19995 & 20896 & -1025 & 155\ ) & \frac{1}{31} & (\ -19995 & 20896 & -1025 & 155\ ) & \\
\frac{5}{6} & \frac{1}{5022} & (\ -469805 & 490960 & -22736 & 5580 & 186\ ) & \frac{1}{5022} & (\ -469805 & 490960 & -22736 & 5580 & 186\ ) & \\
1 & \frac{1}{2604} & (\ 914314 & -955136 & 47983 & -6510 & -558 & 2511\ ) & \frac{1}{2604} & (\ 914314 & -955136 & 47983 & -6510 & -558 & 2511\ ) & \\
\frac{1}{300} & (\ 14 & 0 & 81 & 110 & 0 & 81 & 14\ ) & \frac{1}{300} & (\ 14 & 0 & 81 & 110 & 0 & 81 & 14\ ) & \\
\end{array} \]

A.1-8 RKE(8,11):


Curtis [49] lists a RKE(8,11), listed here as eq. A.75. In eq. A.75 the 'b.'s' are the coefficients w.'s used in this study. Curtis leaves c_2, w_9(b_9), and w_{10}(b_{10}) free; and recommends c_2 = c_3, w_9(b_9) = 1/5, and w_{10}(b_{10}) = 13/80.
Fehlberg [4] lists a RKE[(8,15);(9,17)] with his usual advantage of automatic step-size control and minimum TEB (see section A.1-2). Fourty digit arithmetic is used for the coefficients of this method. This is listed here as eq. A.76, where the 's_i's', B_{ij}'s', and 'c_i's' are the same as the coefficients, c_i, a_{ij} and w_i respectively, used in this study. Also in eq. A.76 i and j take values from 0 to 16 while the formulation used in this study avoids i,j=0. Hence 10 of eq. A.76 is the usual a_{21}, c_0 is w_1 and so on. Naturally a_0(c_1) = 0 in eq. A.76. A formula for calculating the truncation error is also listed.
\[
\begin{align*}
\alpha_1 &= 0.4436 8940 3764 9818 3109 5994 0428 1370 \\
\alpha_2 &= 0.6665 3410 5647 4727 4664 3991 0642 2055 \\
\alpha_3 &= 0.9983 0115 8471 2091 1996 5986 5963 3083 \\
\alpha_4 &= 0.3155 0000 0000 0000 0000 0000 0000 0000 \\
\alpha_5 &= 0.5054 4100 9481 6906 8626 5161 2673 7384 \\
\alpha_6 &= 0.1714 2857 1428 5714 2857 1428 5714 2857 \\
\alpha_7 &= 0.8285 7142 8571 4285 7142 8571 4285 7143 \\
\alpha_8 &= 0.6554 3966 1210 1156 2514 9537 6925 5586 \\
\alpha_9 &= 0.2487 8317 9680 6269 2069 7222 7456 0771 \\
\alpha_{10} &= 0.1090 0000 0000 0000 0000 0000 0000 0000 \\
\alpha_{11} &= 0.8910 0000 0000 0000 0000 0000 0000 0000 \\
\alpha_{12} &= 0.3995 0000 0000 0000 0000 0000 0000 0000 \\
\alpha_{13} &= 0.6005 0000 0000 0000 0000 0000 0000 0000 \\
\alpha_{14} &= 1 \\
\alpha_{15} &= 0 \\
\alpha_{16} &= 1 \\
\beta_{10} &= 0.4436 8940 3764 9818 3109 5994 0428 1370 \\
\beta_{20} &= 0.1663 8352 6411 8681 8666 0997 7860 5514 \\
\beta_{21} &= 0.4991 5057 9235 6015 5986 2993 2981 6541 \\
\beta_{22} &= 0.2495 7528 9617 8022 7999 1496 6490 8271 \\
\beta_{30} &= 0.7487 2586 8853 4068 3997 4489 9472 4812 \\
\beta_{31} &= 0.2066 1891 1634 0060 2426 5567 1043 3165 \\
\beta_{32} &= 0.1770 7880 3779 8634 7040 3809 9728 8319 \\
\beta_{40} &= 0.6819 7715 4138 6949 4669 3770 7681 5048 \times 10^{-1} \\
\beta_{41} &= 0.1092 7823 1526 6649 8227 9038 9092 6157 \\
\beta_{42} &= 0.4021 5962 6423 6799 5421 9805 6369 6087 \times 10^{-2} \\
\beta_{50} &= 0.3921 4118 1690 7898 0444 3923 3017 4325 \\
\beta_{51} &= 0.9889 9281 4091 6465 5304 8447 6543 4355 \times 10^{-1} \\
\beta_{52} &= 0.3513 8370 2279 6396 6951 2044 8735 6703 \times 10^{-2} \\
\beta_{60} &= 0.1247 6099 9831 6001 6621 5206 2587 2489 \\
\beta_{61} &= -0.5574 5546 8349 8979 9643 7429 0146 6348 \times 10^{-1} \\
\beta_{62} &= -0.3680 6865 2862 4220 3724 1531 0108 0691 \\
\beta_{70} &= -0.2227 3897 4694 7600 7645 0240 2094 4166 \times 10^{-1} \\
\beta_{71} &= 0.1374 2908 2567 0291 0729 5636 9124 5744 \times 10^{+1} \\
\beta_{72} &= 0.2049 7390 0271 1160 3002 1593 5409 2206 \times 10^{+1} \\
\beta_{80} &= 0.4546 7962 6413 4715 0077 3519 5060 3349 \times 10^{-1} \\
\end{align*}
\]
\( \beta_{146} = -0.6557 \ 0189 \ 4497 \ 4164 \ 5138 \ 0068 \ 7998 \ 5251 \)
\( \beta_{147} = -0.3908 \ 6144 \ 8804 \ 3986 \ 3435 \ 0255 \ 2024 \ 1310 \)
\( \beta_{148} = 0.2679 \ 4646 \ 7128 \ 5002 \ 2936 \ 5844 \ 2327 \ 1209 \)
\( \beta_{149} = -0.1038 \ 3022 \ 9913 \ 8249 \ 0865 \ 7698 \ 5830 \ 7427 \ \times 10^4 \)
\( \beta_{1410} = 0.1667 \ 2327 \ 3242 \ 5867 \ 1664 \ 7273 \ 4616 \ 8501 \ \times 10^4 \)
\( \beta_{1411} = 0.4955 \ 1925 \ 8553 \ 1597 \ 7067 \ 7329 \ 6707 \ 1441 \)
\( \beta_{1412} = 0.1139 \ 4001 \ 1323 \ 9706 \ 3228 \ 5867 \ 3814 \ 1784 \ \times 10^5 \)
\( \beta_{1413} = 0.5133 \ 6696 \ 4246 \ 5861 \ 3688 \ 1990 \ 9719 \ 1534 \ \times 10^4 \)
\( \beta_{150} = 0.1046 \ 4847 \ 3406 \ 1481 \ 0391 \ 8730 \ 0240 \ 6755 \ \times 10^{-2} \)
\( \beta_{151} = -0.3908 \ 6144 \ 8804 \ 3986 \ 3435 \ 0255 \ 2024 \ 1310 \)
\( \beta_{152} = 0.2679 \ 4646 \ 7128 \ 5002 \ 2936 \ 5844 \ 2327 \ 1209 \)
\( \beta_{153} = -0.1038 \ 3022 \ 9913 \ 8249 \ 0865 \ 7698 \ 5830 \ 7427 \ \times 10^4 \)
\( \beta_{154} = 0.1667 \ 2327 \ 3242 \ 5867 \ 1664 \ 7273 \ 4616 \ 8501 \ \times 10^4 \)
\( \beta_{155} = 0.4955 \ 1925 \ 8553 \ 1597 \ 7067 \ 7329 \ 6707 \ 1441 \)
\( \beta_{156} = 0.1139 \ 4001 \ 1323 \ 9706 \ 3228 \ 5867 \ 3814 \ 1784 \ \times 10^5 \)
\( \beta_{157} = 0.5133 \ 6696 \ 4246 \ 5861 \ 3688 \ 1990 \ 9719 \ 1534 \ \times 10^4 \)

\( c_0 = 0.3225 \ 6083 \ 5002 \ 1624 \ 9913 \ 6129 \ 0096 \ 0247 \ \times 10^{-1} \)
\( c_1 = 0.2598 \ 3725 \ 2837 \ 1540 \ 3018 \ 8870 \ 2317 \ 1963 \)
\( c_2 = 0.9284 \ 7805 \ 9965 \ 7702 \ 7788 \ 0637 \ 1430 \ 2190 \ \times 10^{-1} \)
\( c_3 = 0.1645 \ 2393 \ 5147 \ 6434 \ 2891 \ 6477 \ 3184 \ 2800 \)
\( c_4 = 0.1766 \ 5951 \ 6378 \ 6007 \ 4367 \ 0842 \ 9639 \ 7547 \)
\( c_5 = 0.2392 \ 0102 \ 3203 \ 5275 \ 9374 \ 1089 \ 3332 \ 0941 \)
\( c_6 = 0.3948 \ 4274 \ 6042 \ 0238 \ 3746 \ 7521 \ 1882 \ 9325 \ \times 10^{-2} \)
\( c_7 = 0.3072 \ 6495 \ 4758 \ 6064 \ 0406 \ 3683 \ 0552 \ 2124 \ \times 10^{-1} \)

\( \text{TE} = c_14 (f_0 + f_{14} - f_{15} - f_{16}) h \)
<table>
<thead>
<tr>
<th>(a_{05} = 0.3254)</th>
<th>(b_{05} = 0.2847)</th>
<th>(c_{05} = 0.9783)</th>
<th>(d_{05} = 0.6084)</th>
<th>(e_{05} = 0.5405)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a_{17} = 0.8914)</td>
<td>(b_{17} = 0.7015)</td>
<td>(c_{17} = 0.7774)</td>
<td>(d_{17} = 0.6774)</td>
<td>(e_{17} = 0.6964)</td>
</tr>
<tr>
<td>(a_{26} = 0.8514)</td>
<td>(b_{26} = 0.7114)</td>
<td>(c_{26} = 0.6774)</td>
<td>(d_{26} = 0.6964)</td>
<td>(e_{26} = 0.8853)</td>
</tr>
<tr>
<td>(a_{37} = 0.9783)</td>
<td>(b_{37} = 0.7015)</td>
<td>(c_{37} = 0.7774)</td>
<td>(d_{37} = 0.6774)</td>
<td>(e_{37} = 0.6964)</td>
</tr>
<tr>
<td>(a_{48} = 0.1959)</td>
<td>(b_{48} = 0.7015)</td>
<td>(c_{48} = 0.7774)</td>
<td>(d_{48} = 0.6774)</td>
<td>(e_{48} = 0.6964)</td>
</tr>
<tr>
<td>(a_{59} = 0.4274)</td>
<td>(b_{59} = 0.3254)</td>
<td>(c_{59} = 0.1959)</td>
<td>(d_{59} = 0.7015)</td>
<td>(e_{59} = 0.7774)</td>
</tr>
<tr>
<td>(a_{60} = 0.1743)</td>
<td>(b_{60} = 0.0540)</td>
<td>(c_{60} = 0.5045)</td>
<td>(d_{60} = 0.7015)</td>
<td>(e_{60} = 0.7774)</td>
</tr>
<tr>
<td>(a_{71} = 0.5405)</td>
<td>(b_{71} = 0.2847)</td>
<td>(c_{71} = 0.9783)</td>
<td>(d_{71} = 0.6084)</td>
<td>(e_{71} = 0.5405)</td>
</tr>
<tr>
<td>(a_{82} = 0.1144)</td>
<td>(b_{82} = 0.0540)</td>
<td>(c_{82} = 0.9783)</td>
<td>(d_{82} = 0.6084)</td>
<td>(e_{82} = 0.5405)</td>
</tr>
<tr>
<td>(a_{93} = 0.0277)</td>
<td>(b_{93} = 0.0540)</td>
<td>(c_{93} = 0.9783)</td>
<td>(d_{93} = 0.6084)</td>
<td>(e_{93} = 0.5405)</td>
</tr>
<tr>
<td>(a_{104} = 0.1479)</td>
<td>(b_{104} = 0.0540)</td>
<td>(c_{104} = 0.9783)</td>
<td>(d_{104} = 0.6084)</td>
<td>(e_{104} = 0.5405)</td>
</tr>
<tr>
<td>(a_{115} = 0.0353)</td>
<td>(b_{115} = 0.0540)</td>
<td>(c_{115} = 0.9783)</td>
<td>(d_{115} = 0.6084)</td>
<td>(e_{115} = 0.5405)</td>
</tr>
<tr>
<td>(a_{126} = 0.0277)</td>
<td>(b_{126} = 0.0540)</td>
<td>(c_{126} = 0.9783)</td>
<td>(d_{126} = 0.6084)</td>
<td>(e_{126} = 0.5405)</td>
</tr>
<tr>
<td>(a_{137} = 0.0114)</td>
<td>(b_{137} = 0.0540)</td>
<td>(c_{137} = 0.9783)</td>
<td>(d_{137} = 0.6084)</td>
<td>(e_{137} = 0.5405)</td>
</tr>
<tr>
<td>(a_{148} = 0.0053)</td>
<td>(b_{148} = 0.0540)</td>
<td>(c_{148} = 0.9783)</td>
<td>(d_{148} = 0.6084)</td>
<td>(e_{148} = 0.5405)</td>
</tr>
<tr>
<td>(a_{159} = 0.0053)</td>
<td>(b_{159} = 0.0540)</td>
<td>(c_{159} = 0.9783)</td>
<td>(d_{159} = 0.6084)</td>
<td>(e_{159} = 0.5405)</td>
</tr>
</tbody>
</table>

**Fehlberg**

**A.76 contd.**
Shanks [27] has listed two 'almost' eighth order formulae (see section A.1-5), one a RKE(8,10), and the other a RKE(8,12). As expected the (8,10) is faster, while the (8,12) is more accurate (not much more accurate, as his own tests prove).

<table>
<thead>
<tr>
<th>0</th>
<th>( \frac{4}{27} )</th>
<th>( \frac{4}{27} )</th>
<th>( \frac{4}{27} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>2/9</td>
<td>( \frac{1}{18} )</td>
<td>(1 3)</td>
<td>( \frac{1}{18} )</td>
</tr>
<tr>
<td>1/3</td>
<td>( \frac{1}{12} )</td>
<td>(1 0 3)</td>
<td>( \frac{1}{12} )</td>
</tr>
<tr>
<td>1/2</td>
<td>( \frac{1}{8} )</td>
<td>(1 0 0 3)</td>
<td>( \frac{1}{8} )</td>
</tr>
<tr>
<td>2/3</td>
<td>( \frac{1}{54} )</td>
<td>(13 0 27 42 8)</td>
<td>( \frac{1}{54} )</td>
</tr>
<tr>
<td>1/6</td>
<td>( \frac{1}{4320} )</td>
<td>(109 0 -54 966 -824 243)</td>
<td>( \frac{1}{4320} )</td>
</tr>
<tr>
<td>1/20</td>
<td>( \frac{1}{20} )</td>
<td>(-231 0 81 -1164 656 -122 800)</td>
<td>( \frac{1}{20} )</td>
</tr>
<tr>
<td>1/288</td>
<td>( \frac{1}{288} )</td>
<td>(-127 0 18 -678 456 -9 576 4)</td>
<td>( \frac{1}{288} )</td>
</tr>
<tr>
<td>1/820</td>
<td>( \frac{1}{820} )</td>
<td>(1481 0 -81 7104 -3376 72 -5040 -60 720)</td>
<td>( \frac{1}{820} )</td>
</tr>
<tr>
<td>1/840</td>
<td>( \frac{1}{840} )</td>
<td>(41 0 0 27 272 27 216 0 216 41)</td>
<td>( \frac{1}{840} )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>0</th>
<th>( \frac{1}{9} )</th>
<th>( \frac{1}{9} )</th>
<th>( \frac{1}{9} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/9</td>
<td>( \frac{1}{27} )</td>
<td>(1 3)</td>
<td>( \frac{1}{27} )</td>
</tr>
<tr>
<td>1/13</td>
<td>( \frac{1}{13} )</td>
<td>(1 0 3)</td>
<td>( \frac{1}{13} )</td>
</tr>
<tr>
<td>3/10</td>
<td>( \frac{3}{90} )</td>
<td>(19 0 33 -12)</td>
<td>( \frac{3}{90} )</td>
</tr>
<tr>
<td>1/6</td>
<td>( \frac{1}{972} )</td>
<td>(33 0 0 4 125)</td>
<td>( \frac{1}{972} )</td>
</tr>
<tr>
<td>1/35</td>
<td>( \frac{1}{35} )</td>
<td>(-21 0 0 76 125 -162)</td>
<td>( \frac{1}{35} )</td>
</tr>
<tr>
<td>1/243</td>
<td>( \frac{1}{243} )</td>
<td>(-30 0 0 -32 125 0 99)</td>
<td>( \frac{1}{243} )</td>
</tr>
<tr>
<td>1/324</td>
<td>( \frac{1}{324} )</td>
<td>(1175 0 0 -3456 -6250 8424 262 -27)</td>
<td>( \frac{1}{324} )</td>
</tr>
<tr>
<td>1/324</td>
<td>( \frac{1}{324} )</td>
<td>(293 0 0 -852 -1275 1836 -118 162 324)</td>
<td>( \frac{1}{324} )</td>
</tr>
<tr>
<td>1/1620</td>
<td>( \frac{1}{1620} )</td>
<td>(1303 0 0 4240 -6875 9990 1030 0 0 162)</td>
<td>( \frac{1}{1620} )</td>
</tr>
<tr>
<td>1/4420</td>
<td>( \frac{1}{4420} )</td>
<td>(-8595 0 0 30720 48750 -66096 378 -729 -1944 -1256 3240)</td>
<td>( \frac{1}{4420} )</td>
</tr>
</tbody>
</table>
Increasing use of higher precision arithmetic is required, to realize the advantages of this order over the seventh order (slim advantage in accuracy)—paying the price of increased computational time.

A.1-9 RKE's of Ninth and Higher Order

The deriving equations for ninth and higher orders are listed, but not solved in Rosen [5].

Shanks is supposed to have developed (not yet published) ninth and tenth order RKE's.

There is no real reason for pushing on for orders beyond the eighth (except for special problems). According to Fehlberg [4] accuracy hardly improves after the seventh order methods.

Sarafyan is working on a computer formulation and solution, of the deriving equations leading to high order methods (see Sarafyan and Brown [7]); however the results are yet to be published.

A.2 Semi-Implicit Runge-Kutta Methods - RKSI(p,V):

It is easier to use the autonomous form of an ordinary differential equation system when dealing with these types of methods.

\[
\frac{dY}{dT} = F(T,Y)
\]  \hspace{1cm} A.79

The explicit dependence of T in F(T,Y) can be removed by making T the (n+1)th component of the n component vector Y. This makes the system n+1 dimensional rather than n dimensional.

let \[ Y_{n+1} = T \]
\[ \frac{dY_{n+1}}{dT} = 1 \]
\[ \frac{dY}{dT} = F(Y) \]  

\[ A.80 \]

where:

\[ \begin{pmatrix} Y_1 \\ \vdots \\ Y_{n+1} \end{pmatrix} = \begin{pmatrix} F_1(Y_1, \ldots, Y_n, Y_{n+1}) \\ \vdots \\ F_n(Y_1, \ldots, Y_n, Y_{n+1}) \end{pmatrix} \]

The initial conditions would also be accordingly altered, i.e., \( Y_{n+1}(T_0) = T_0 \). There is no need however to solve the \( n+1 \) component equation of eq. A.80 as its solution is always \( Y_{n+1} = T \).

When a Runge-Kutta method is used on this autonomous form, the increments coefficients of the abcissa, i.e., the c's, become a's, i.e., the increment coefficients of the vector component \( Y_{n+1} \).

In Chapter 2, section 2.2 the method of solving for each \( k_i \) of the semi-implicit form was discussed; this involved a Jacobian inversion. Restating eq. 2.33 in matrix form to apply to a system of equations, with the \( a_{ij}'s \) involved in the Jacobian written as \( a_{ij}'a \).

\[ k_i = [I - h a_{ii} F_y(Y_n + \sum_{j=1}^{i-1} a_{ij}' k_j)^{-1} - h F(Y_n + \sum_{j=1}^{i-1} a_{ij} k_j) \]  

\[ A.81 \]

The RKSI type method used to solve eq. A.80 is:
\[ Y_{n+1} = Y_n + \sum_{i=1}^{V} w_i k_i \]

\[ k_i = hF(Y_n + \sum_{j=1}^{i} a_{ij} k_j) \]

To solve for each \( k_i \), the formulation of eq. A.81 is used.

The coefficient matrix is of the form:

\[
\begin{pmatrix}
  a_{11} & a_{21} & \cdots & a_{12} & \cdots & a_{1v} \\
  a_{21} & a_{22} & \cdots & a_{22} & \cdots & a_{2v} \\
  \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
  a_{11} & a_{12} & \cdots & a_{ii} & \cdots & a_{i1} \\
  \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
  a_{v1} & a_{v2} & \cdots & a_{vv} & \cdots & a_{v1} \\
  w_1 & w_2 & \cdots & w_1 & \cdots & w_v
\end{pmatrix}
\]

Allen [50], Rosenbrock [51] and Haines [52] list the deriving equations. Allen [50] gives a thorough discussion of the deriving equations and the stability advantages of RKSI's over RKE's; and also gives coefficients of the truncation errors. In general RKSI's require less stages, \( V \) per step as compared to RKE's, to achieve a \( p \) order method. The price paid by RKSI's, for achieving extended stability as compared to RKE's, is the additional computational effort required to compute Jacobians and their inverses; but this price is not as steep as the one paid by RKI's, where iteration for \( k_i \)'s is required.

Lapadis and Seinfeld [6] recommend the use of RKSI only for those cases where stability is critical, for example, with still systems; because in general RKSI's are not significantly more accurate
than comparable order RKE's; and RKSI's usually require more computational time as compared to RKE's.

The following sections are titled RKSI(p) instead of RKSI(p,V).

A.2-1 RKSI(1):

Allen [50] states that no such case of interest exists.

A.2-2 RKSI(2):

Allen [50] and Rosenbrock [51] list the deriving equations and solve them. Allen [50] gives a V=2 method while Rosenbrock gives a V=2 process.

\[
\begin{array}{c|c|c}
1/2 & 0 & \text{Allen A.83} \\
1 & & \\
\end{array}
\]

\[
\begin{array}{c|c|c}
1 - \frac{\sqrt{2}}{2} & 0 & \text{Rosenbrock A.84} \\
\frac{\sqrt{2} - 1}{2} & 1 - \frac{\sqrt{2}}{2} & \\
0 & 1 & \\
\end{array}
\]

Allen [50] however admits that his method, eq. A.83 does not possess significant stability advantages over a second order RKE.

A.2-3 RKSI(3):

The deriving equations are listed and solved in Allen [50], Rosenbrock [51] and Haines [52].

Allen [50] lists three methods with V=2:

\[
\begin{array}{c|c|c|c}
0 & 1/3 & \text{Allen A.85} \\
0 & 1/3 & \\
-1/2 & 3/2 & \\
\end{array}
\]
Eq. A.87 is the best such method as judged by various tests, conducted in Allen [50].

Rosenbrock [51] lists a method with $V=2$:

| 1.40824829 | 0 | Rosenbrock A.88 |
| 0.17378667 | 0.17378667 | .59175171 |
| 0.17378667 | 0.17378667 | .59175171 |

Calahan [53] also lists a $V=2$ method, which is the same as Allen's eq. A.87.

Haines [52] derives a $V=4$ method, the extra stages per step being used to give a truncation error estimate and hence allow step size control. This method consists of two equations, each with $V=4$ stages. These two equations are compared to give a truncation error estimate. Both these equations have mostly common coefficients. This method is listed in two parts because each part can be used independently to give a third order method, or in combination to give an error estimate.
Error_n = (29/9)[(Y_n with eq. A.89) - (Y_n with eq. A.90)]

Haines [52] also compares his method, which combines eqs. A.89 and A.90 to give step size control, with a RKE(4,4) which uses a similar step size control. His results point out that his method is superior on basis of computing time and stability.

A.2-4 RKSI(4):

Allen [50] lists thirth fourth order methods with V=3. Three such methods are presented here, these three were judged in Allen [50], and found to be superior to the rest of his methods on some points.
| 0.28915127 | 0.28915127 | 0 |
| 0.57118895 | 0.57118895 | 0 |
| 0.53873135 | -0.024693209 | 0.33215386 |
| 0.72862466 | 4.21831023 | -3.94693489 |

| 0.86234511 | 0.86234511 | 0 |
| -1.28325229 | 1.23119918 | 0 |
| -0.80332019 | -0.091154713 | 1.23119818 |
| -0.32266573 | -0.85621119 | 2.17887692 |

Allen (largest region of stability)

Allen (best for numerical stability)

Butcher [2] has listed a RKSI(4,3); his method does not apply to the autonomous form of the differential equations, but to the usual form used throughout this study. (The method could easily be converted to apply to an autonomous form by simply removing the $c_i$'s.)

| 0 | 0 |
| 1/2 | 1/4 | 1/4 |
| 1 | 0 | 1 | 0 |
| 1/6 | 2/3 | 1/6 |

Butcher

A.3 Implicit Runge-Kutta Methods - RKI(p,V):

Restating the equations for a RKI(p,V):

$$Y_{n+1} = Y_n + \sum_{i=1}^{V} w_ik_i$$

The coefficient matrix for these types of methods is full; and hence each $k_i$ has to be solved for iteratively at each integration step.
The use of these RKI's is recommended for two situations: one where stability is critical and a self-starting process is desired; two when the differential equation system is such that the cost of computing \( f(T,Y) \) for a given value of \( T \) is high compared to the cost of repeating this computation with \( Y \) changed but \( T \) unchanged. In the second situation the iterative nature of an RKI would not be an objection; especially if the differential equation system is linear, then the iterations can be replaced by standard linear algebra techniques.

**Advantages:** Stability characteristics better than comparable order RKSI's or RKE's. Always stable for any \( h \).

Higher \( p \) for a particular \( V \) as compared to RKSI's and RKE's. The deriving equations are easier to formulate than those of RKI's and RKE's.

**Disadvantages:** For even moderately complex differential equation systems the iterative solution of \( k_i \)'s requires more computational effort than comparable order semi-implicit or explicit methods.

The coefficient matrix will be presented in the following form:

\[
\begin{align*}
\mathbf{c}_1 & \left| a_{11} \ldots a_{1j} \ldots a_{1v} \\
\vdots & \vdots \\
\mathbf{c}_i & \left| a_{i1} \ldots a_{ij} \ldots a_{iv} \\
\vdots & \vdots \\
\mathbf{c}_v & \left| a_{v1} \ldots a_{vj} \ldots a_{vv} \\
\hline
\mathbf{w}_1 & \ldots w_j \ldots w_v
\end{align*}
\]

Most currently available implicit processes were derived by Butcher. The theory behind implicit methods, attainable order, etc.,
is dealt in Butcher [1] and Verner [54]. Deriving equations and implicit methods are listed in Butcher [2], Butcher [3] and Butcher [55]. Lapidus and Seinfeld [6] also list some of these Butcher methods.

Implicit Runge-Kutta methods are based on quadratic formulae (The Taylor Series analysis can also be used to derive implicit methods but the quadratic approach is simpler). Basically Implicit Runge-Kutta's can be divided into three classes of methods: Gauss-Legendre quadratic forms, Radau quadratic forms, and Lobatto quadratic forms.

<table>
<thead>
<tr>
<th>Quadrature Form</th>
<th>Abcissa points (c_i's) specified</th>
<th>V</th>
<th>p</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gauss-Legendre</td>
<td>All c_i's found as roots of a Legendre polynomial, no c_i specified</td>
<td>V</td>
<td>2V</td>
</tr>
<tr>
<td>Radau_1</td>
<td>c_1 = 0 specified</td>
<td>V</td>
<td>(2V-1)</td>
</tr>
<tr>
<td>Radau_II</td>
<td>c_V = 1 specified</td>
<td>V</td>
<td>(2V-1)</td>
</tr>
<tr>
<td>Lobatto</td>
<td>c_1 = 0 and c_V = 1 specified</td>
<td>V</td>
<td>(2V-2)</td>
</tr>
</tbody>
</table>

when c_1 = 0 \quad a_{11} = a_{12} = a_{13} = \ldots = a_{1V} = 0

when c_V = 1 \quad a_{1V} = a_{2V} = a_{3V} = \ldots = a_{VV} = 0

Hence the Lobatto forms have the first row and last column of the coefficient matrix as zeros. Thus as c_1 = 0 and c_V = 1 were specified two k_i's, k_1 and k_V are available explicitly and so can be solved for by iteration. The Radau_1 and Radau_II forms have only one k_i available explicitly as only one c_i is specified. While the Gauss-Legendre forms have no k_i available explicitly as no c_i is
specified. Because of the number of abcissa points specified, the Gauss-Legendre forms have the highest p for a given V, while Radau forms are next and Lobatto forms last.

Butcher [2] also suggests a convergent iterative procedure applicable for calculating the implicit $k_i$'s of any RKI.

If $n$ is the iteration count for a particular $k_i$ the convergent iterative procedure is:

$$k_i^{(n)} = h j (T_n + c_i h, Y_n + \sum_{j=1}^{i-1} a_{ij} k_j^{(n)} + \sum_{j>i} a_{ij} k_j^{(n-1)})$$

Butcher proves that this procedure is convergent.

A.3-1 RKI$(p,V)$'s of the Gauss-Legendre Quadrative forms:

These kinds of RKI's are listed in Butcher [2] from $p = 2$ to $p = 10$ ($p = 2V$), an error analysis is also included.

$$\begin{array}{c|ccc}
1/2 & 1/2 & \text{RRI}(2,1) \\
\hline
1 \\
\end{array}
\begin{array}{c|ccc}
1/2 - \sqrt{3}/6 & 1/4 & 1/4 - \sqrt{3}/6 & \text{RRI}(4,2) \\
1/2 + \sqrt{3}/6 & 1/4 + \sqrt{3}/6 & 1/4 \\
\hline
1/2 & 1/2 \\
\end{array}$$

A.96

A.97
\[
\begin{array}{c|cccc}
\sqrt{15} & 5 & 2 - \sqrt{15} & 5 & - \sqrt{15} \\
10 & 5 & 9 & - \sqrt{15} & 35 & - \frac{\sqrt{15}}{30} \\
\hline
\frac{1}{2} & 5 & \frac{9}{10} & 10 & 35 & - \frac{\sqrt{15}}{30} \\
\frac{1}{2} + \sqrt{15} & 5 & \frac{9}{35} & \frac{5}{10} & \frac{9}{30} & 5 \\
\end{array}
\]

\[A.98\]

\[
\begin{array}{c|ccccc}
\sqrt{15} & 5 & 2 - \sqrt{15} & 5 & - \sqrt{15} \\
10 & 5 & 9 & - \sqrt{15} & 35 & - \frac{\sqrt{15}}{30} \\
\hline
\frac{1}{2} & 5 & \frac{9}{10} & 10 & 35 & - \frac{\sqrt{15}}{30} \\
\frac{1}{2} + \sqrt{15} & 5 & \frac{9}{35} & \frac{5}{10} & \frac{9}{30} & 5 \\
\end{array}
\]

\[A.99\]

\[
\begin{array}{c|ccccc}
\sqrt{15} & 5 & 2 - \sqrt{15} & 5 & - \sqrt{15} \\
10 & 5 & 9 & - \sqrt{15} & 35 & - \frac{\sqrt{15}}{30} \\
\hline
\frac{1}{2} & 5 & \frac{9}{10} & 10 & 35 & - \frac{\sqrt{15}}{30} \\
\frac{1}{2} + \sqrt{15} & 5 & \frac{9}{35} & \frac{5}{10} & \frac{9}{30} & 5 \\
\end{array}
\]

\[A.100\]
A.3-2 RKI(p,V)’s of the Radau Quadrature forms:

These types of RKI’s are listed in Butcher [55], from p=1 to p=5 \( (p = 2V - 1) \). Butcher [55] also describes an ingenious error analysis applicable to the Radau and Lobatto (see section A.3-3) forms of RKI’s.

\[
\begin{array}{c|cccc}
0 & 0 & 0 & 0 & 0 \\
\hline
1 & & & & \\
\end{array}
\]

RKI(1,1) 
A.101

\[
\begin{array}{c|cccc}
0 & 0 & 0 & 0 & 0 \\
\hline
2/3 & 1/3 & 1/3 & \\
\hline
1/4 & 3/4 & \\
\end{array}
\]

RKI(3,2) 
A.102

\[
\begin{array}{c|cccc}
0 & 0 & 0 & 0 & 0 \\
\hline
6 - \sqrt{6}/10 & 9 + \sqrt{6}/75 & 24 + \sqrt{6}/120 & 168 - 73\sqrt{6}/600 & \text{RKI}(5,3) \\
\hline
6 + \sqrt{6}/10 & 9 - \sqrt{6}/75 & 168 + 73\sqrt{6}/600 & 24 - \sqrt{6}/120 & \\
\hline
1 & 16 + \sqrt{6}/36 & 16 - \sqrt{6}/36 & \\
\end{array}
\]

RKI(3,2) 
A.103

\[
\begin{array}{c|cccc}
1/3 & 1/3 & 0 & 0 \\
\hline
1 & 1 & 0 & \\
\hline
3/4 & 1/4 & \\
\end{array}
\]

RKI(5,3) 
A.104

\[
\begin{array}{c|cccc}
4 - \sqrt{6}/10 & 24 - \sqrt{6}/120 & 24 - 11\sqrt{6}/120 & 0 & \text{RKI}(5,3) \\
\hline
4 + \sqrt{6}/10 & 24 + 11\sqrt{6}/120 & 24 + \sqrt{6}/120 & 0 & \\
\hline
1 & 6 - \sqrt{6}/12 & 6 + \sqrt{6}/12 & 0 & \\
\hline
16 - \sqrt{6}/36 & 16 + \sqrt{6}/36 & 1/9 & \\
\end{array}
\]

RKI(5,3) 
A.105
A.3-3  RKI(p,v)'s of the Lobatto Quadrative forms:

Butcher [55] lists these types of RKI's from p=2 to p=12
(p = 2v - 2). An ingenious error analysis applicable to these forms
is also described.

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>RKI(2,2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>A.106</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1/2</td>
<td>1/2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

|          | 0        | 0        | 0        | RKI(4,3) |          |          |
| 1/2      | 1/4      | 1/4      | 0        | A.107    |          |          |
| 1        | 0        | 1        | 0        |          |          |          |
|          | 1/6      | 2/3      | 1/6      |          |          |          |

|          | 0        | 0        | 0        | 0        | 0        | 0        |
| 5 - √5  | 10       | 5 + √5   | 10       | 15 - 7 √5 | 60      | 0        |
| 5 + √5  | 10       | 5 - √5   | 15 + 7 √5 | 1        | 60      | 0        |
| 1        | 6        | 1        | 5 - √5   | 5 + √5   | 12      | 0        |
|          | 1        | 5        | 5        | 1        | 12      | 12       |

|          | 0        | 0        | 0        | 0        | 0        | 0        |
| 7 - √21  | 14       | 1        | 13 - 3 √21 | 14 - 3 √21 | 0        |
| 1        | 9        | 14 + 21 √21 | 11      | 91 - 21 √21 | 0        |
|          | 32       | 14       | 126      | 3       | 0        |
| 7 + √21  | 14       | 14 + 3 √21 | 13 + 3 √21 | 1        | 0        |
| 1        | 0        | 7        | 2        | 7        | 0        |
|          | 1        | 49       | 16       | 49       | 1        |
|          | 1/20     | 490      | 45       | 180      | 20       |
\[
\begin{array}{cccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1-\omega & 2 & 1 - \omega & 1 - \omega & 0 & 0 & 0 & 0 \\
1-\omega' & 2 & 1 - \omega' & 1 - \omega' & 0 & 0 & 0 & 0 \\
1+\omega' & 2 & 1 + \omega' & 1 + \omega' & 0 & 0 & 0 & 0 \\
1+\omega & 2 & 1 + \omega & 1 + \omega & 0 & 0 & 0 & 0 \\
1 & 30 & 2m & 2m' & 2m' & 2m' & 2m & 1 \\
\end{array}
\]

\[
\begin{align*}
\omega &= \frac{14 - \sqrt{2}}{120}, \quad \omega' = \frac{14 + \sqrt{2}}{120}, \\
\omega'' &= \sqrt{\frac{7 + 2\sqrt{2}}{21}}, \quad \omega''' = \sqrt{\frac{7 - 2\sqrt{2}}{21}}, \\
\omega(\text{II}) &= \frac{7 - \sqrt{2}}{120}, \quad \omega'(\text{II}) = \frac{7 + \sqrt{2}}{120}, \\
\omega(\text{III}) &= \frac{14 + \sqrt{2}}{\sqrt{21}}, \quad \omega'(\text{III}) = \frac{14 - \sqrt{2}}{\sqrt{21}}, \\
\omega(\text{IV}) &= \frac{22 + 17\sqrt{2}}{300}, \quad \omega'(\text{IV}) = \frac{22 - 17\sqrt{2}}{300}, \\
\omega(\text{V}) &= \frac{22 - 5\sqrt{2}}{60}, \quad \omega'(\text{V}) = \frac{22 + 5\sqrt{2}}{60}, \\
\end{align*}
\]

\[
\begin{array}{cccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1-\omega & 2 & 1 - \omega & 1 - \omega & 0 & 0 & 0 & 0 \\
1-\omega' & 2 & 1 - \omega' & 1 - \omega' & 0 & 0 & 0 & 0 \\
1+\omega' & 2 & 1 + \omega' & 1 + \omega' & 0 & 0 & 0 & 0 \\
1+\omega & 2 & 1 + \omega & 1 + \omega & 0 & 0 & 0 & 0 \\
1 & 42 & 32 & 126 & 25 & 126 & 25 & 1 \\
\end{array}
\]

\[
\begin{align*}
\omega &= \frac{124 - 7\sqrt{15}}{1400}, \quad \omega' = \frac{124 + 7\sqrt{15}}{1400}, \\
\omega'' &= \sqrt{\frac{15 + 2\sqrt{15}}{33}}, \quad \omega''' = \sqrt{\frac{15 - 2\sqrt{15}}{33}}, \\
\omega(\text{II}) &= \frac{413}{360}, \quad \omega'(\text{II}) = \frac{413}{360}, \\
\omega(\text{III}) &= \frac{54 + \sqrt{15}}{2178}, \quad \omega'(\text{III}) = \frac{54 - \sqrt{15}}{2178}, \\
\omega(\text{IV}) &= \frac{96 + 27\sqrt{15}}{1230}, \quad \omega'(\text{IV}) = \frac{96 - 27\sqrt{15}}{1230}, \\
\omega(\text{V}) &= \frac{3270 + 100\sqrt{15}}{27225}, \quad \omega'(\text{V}) = \frac{3270 - 100\sqrt{15}}{27225}, \\
\omega'' &= \frac{26 + \sqrt{15}}{1035}, \quad \omega'(\text{V}) = \frac{26 - \sqrt{15}}{1035}, \\
\omega(\text{V}) &= \frac{8 + \sqrt{15}}{60}, \quad \omega'(\text{V}) = \frac{8 - \sqrt{15}}{60}, \\
\omega(\text{V}) &= \frac{111 + 7\sqrt{15}}{840}, \quad \omega'(\text{V}) = \frac{111 - 7\sqrt{15}}{840}, \\
\end{align*}
\]

\[\text{RKI}(10,6)\]
\[\text{RKI}(12,7)\]

\[A.110\]
\[A.111\]
A.4 Multistep Runge-Kutta Methods - RKMS(p,V):

One way of reducing the stages V of a p order Runge-Kutta is to use an implicit Runge-Kutta—the price paid would be the iterations required. Another way of reducing V for a given p order Runge-Kutta is to utilize the solutions of one or more previously evaluated steps. This result is a Multistep Runge-Kutta. The price paid here is that the inherent self-starting, easily instatuteable step size change, accuracy, and stability advantages of the single step Runge-Kutta's are lost. Another disadvantage (trivial) of a multistep method, compared to a single step method, is that extra storage is required to save the solutions of previously evaluated steps. The typical complex error analysis of single step Runge-Kutta's occurs in RKMS's too.

On the whole standard multistep predictor-corrector methods perform better than RKMS's; so there does not seem much point in using a RKMS instead of a RKE, if a multistep method is desired, a standard predictor-corrector is recommended.

Based on the usual formulation of a Runge-Kutta method, applicable to a first order ODE, a RKMS(p,V) is defined by:

\[
Y_{n+1} = Y_n + \sum_{s=1}^{V} \left[ \sum_{i=1}^{S} w_i^{n-s+1} k_i^{n-s+1} \right]
\]

\[
k_i^{n-s+1} = h j \left( T_{n-s+1} + c_i^{n-s+1} h, y_{n-s+1} + \sum_{j=1}^{M_i} a_{ij}^{n-s+1} k_j^{n-s+1} \right)
\]

\[
S_n \leq n+1
\]

\[
M_i \leq V
\]

\[
n = 1, 2, 3, \ldots
\]
The superscripts in eq. A.112 are used to identify \( w_i \), \( k_i \), \( c_i \) and \( a_{ij} \) at a particular time step. For example, \( S = 1 \) makes eq. A.112 a single step Runge-Kutta; \( S = 2 \) makes eq. A.112 a two step Runge-Kutta, and the \( k_i \)'s of the \((n-1)\)th time step are used together with the \( k_i \)'s of the \(n\)th time step, to arrive at the solution \( Y_{n+1} \). If \( S = n+1 \), then the solutions of all previously evaluated steps, including the initial conditions; are used to calculate \( Y_{n+1} \). Henceforth \( RKMS(p,V,S_n) \) will be used to designate these methods. The value of \( S_n \) identifies the number of steps used in the RKMS. Bryne and Lambert 59 show that there is not much point in going beyond a two step \( (S = 2) \) Runge-Kutta, because both accuracy and stability of these RKMS's deteriorate as compared to single step Runge-Kutta's.

The rest of this section will discuss \( S = 2 \), or two step RKMS's, eq. A.112 is reformulated in a more convenient form for \( S = 2 \).

\[
\begin{align*}
Y_{n+1} &= Y_n + \sum_{i=1}^{V} u_i \delta_i + \sum_{i=1}^{V} w_i k_i \\
\delta_i &= hf(T_{n-1} + b_i h, Y_{n-1} + \sum_{j=1}^{M_i} d_{ij} \delta_i) \\
k_i &= hf(T_n + c_{ij}, Y_n + \sum_{j=1}^{M_i} a_{ij} k_j) \\
M_i &\leq V
\end{align*}
\]

Usually \( M_i = 1 - 1 \) is used to give an explicit formulation.

Rosen [62] gives the deriving equations for \( RKMS(4,3,2) \) and \( RKMS(5,5,2) \) using the convenient quadratic approach (convenient as compared to the Taylor Series approach). Butcher [56,57] also gives the deriving equation, but the methods of these papers are really hybrid methods, not RKMS's, and so come under section A.5.
Bryne and Lambert [59] present the deriving equations for two RKM's: a RKMS(3,2,2) and a RKMS(4,3,2). The general solution, error analysis and proof of convergence is also presented. Bryne [58] gives the parameters which make the above two methods optimum (minimum TEB based on the Ralston [8] criteria, see sections A.1-3 and A.1-4). Lapidus and Seinfeld [6] also list these two methods.

RKMS(3,2,2):

\[
\begin{align*}
\begin{array}{l}
\text{n-1} \\
b_1 &= 0 \\
b_2 &= c_2 \\
d_{21} &= c_2 \\
\end{array}
\quad
\begin{array}{l}
\text{n} \\
c_1 &= 0 \\
c_2 &= c_2 \\
a_1 &= c_2 \\
\end{array}
\]

\[
\begin{align*}
\begin{array}{l}
u_1 &= 1-w_1 \\
u_2 &= -w_2 \\
\end{array}
\quad
\begin{array}{l}
w_1 &= \frac{(18c_2-5)}{12c_2} \\
w_2 &= \frac{5}{12c_2} \\
\end{array}
\]

\(c_2\) free, \(c_2 = \frac{4}{5}\) gives an optimum method.

RKMS(4,3,2):

\[
\begin{align*}
\begin{array}{l}
\text{n-1} \\
b_1 &= 0 \\
b_2 &= c_2 \\
b_3 &= c_3 \\
d_{21} &= c_2 \\
da_{31} &= c_3 \\
\end{array}
\quad
\begin{array}{l}
\text{n} \\
c_1 &= 0 \\
c_2 &= c_2 \\
c_3 &= c_3 \\
a_{21} &= c_2 \\
a_{31} &= c_3-a_{32} \\
\end{array}
\]

\[
\begin{align*}
\begin{array}{l}
u_1 &= 1-w_1 \\
u_2 &= -w_2 \\
u_3 &= -w_1 \\
\end{array}
\quad
\begin{array}{l}
w_1 &= \frac{[2c_3(c_3-c_2)]}{12c_2c_3} \\
w_2 &= \frac{[4+18c_2c_3-5(c_2+c_3)]}{[4-5c_3]} \\
w_3 &= \frac{[12c_3(c_2-c_3)]}{[5c_2-4]} \\
\end{array}
\]

\(c_2\) and \(c_3\) are free, \(c_2 = .541\) and \(c_3 = .722779927\) give an optimum method.
Lapidus and Seinfeld also list the above two methods.

These two methods, eq. A.114 and A.115, are not as accurate as comparable order RKE's (though they are faster).

Chai develops an optimum RKMS(4,3,2) using an ingenious derivation. Chai notes that if an RKE(4,4) has \( w_2 = 0 \) then the \( k_2 \) can be reformulated in a multistep form without changing the \( c_i \)'s, \( a_{ij} \)'s, and \( w_i \)'s of the RKE(4,4). The method is hence defined by the usual RKE(4,4) formulation but with \( k_2 \) given differently.

\[
k_2 = hf(T_n, Y_n) + h^2 c_2 f'(T_n, Y_n) \quad A.116
\]

\[
+ \frac{h^3}{2!} e_1 j'''(T_n, Y_n) + \frac{h^4}{3!} e_2 f'''(T_n, Y_n) + \cdots
\]

where

\[ c_2 \neq 0 \]

\[ e_1 \text{ and } e_2 \text{ are constants.} \]

By using a finite difference approximation for \( f'(T_n, Y_n) \) in eq. A.116 Chai arrives at:

\[
k_2 = q_1 k_1 + q_2 VY_{n-1} + q_3 g_1 \quad A.117
\]

where:

\[ VY_{n-1} = Y_n - Y_{n-1} \]

\[ VY_{n-1} = \sum_{i=1}^{V} w_i g_i \quad A.118 \]

the \( g_i \)'s being the \( k_i \)'s of the previous step.

Thus \( V=3 \) is achieved because no function evaluation is necessary for \( k_2 \).
By comparing a Taylor series expansion of eq. A.117, up to second order terms, with the first two terms of eq. A.116, gives a set of conditions for the q's.

\[ q_1 + q_2 + q_3 = 1 \]
\[ \frac{q_2}{2} + q_3 = -c_2 \]

Thus one q is arbitrary.

An optimum RKMS(4,3,2) is hence formulated by Chai (minimum TEB) this uses the same c_i's, a_ij's and w_i's as the RKE(4,4) from which it is derived, hence no separate starting method is necessary.

**RKMS(4,3,2):**

- \( c_1 = 0 \)
- \( c_2 = c_2 \quad q_1 = 1 + \frac{26}{5} c_2 \quad q_2 = -\frac{42}{5} c_2 \quad q_3 = \frac{16}{5} c_2 \)
- \( c_3 = 1/2 \)
- \( c_4 = 1 \)
- \( a_{21} = c_2 \)
- \( a_{31} = \frac{1}{2} - \frac{1}{8c_2} \quad a_{32} = \frac{1}{8c_2} \)
- \( a_{41} = \frac{1}{2c_2} - 1 \quad a_{42} = -\frac{1}{2c_2} \quad a_{43} = 2 \)
- \( w_1 = 1/6 \quad w_2 = 0 \quad w_3 = \frac{2}{3} \quad w_4 = 1/6 \)
- \( c_2 = 1/2 \) or \( 1/4 \) is recommended.

Chai's tests show that this RKMS is superior on accuracy and time to the RKE with which it shares its coefficients. An error expression for correcting the solution, to yield a fifth order accuracy is also given. Due to its simplicity of application and other advantages, this method is highly recommended.

Gruttke [61] gives the deriving equations for a RKMS(5,4,2). These equations are solved to yield various methods which are
extensively tested. An optimum based on the Ralston [8] criteria was
determined.

RKMS(5,4,2):

\[
\begin{align*}
    n-1 & \quad n \\
    b_1 = 0 & \quad c_1 = 0 \\
    b_2 = c_2 & \quad c_2 = c_2 \\
    b_3 = c_3 & \quad c_3 = c_3 \\
    b_4 = c_4 & \quad c_4 = 62 \quad 85 \\
    d_{21} = c_2 & \quad a_{21} = c_2 \\
    d_{31} = a_{31} & \quad a_{31} = c_3 - a_{32} \\
    d_{32} = a_{32} & \quad a_{32} = \frac{31 - 40c_4}{240 w_3 c_2 (c_3 - c_4)} \\
    d_{41} = a_{41} & \quad a_{41} = c_4 - a_{42} - a_{43} \\
    d_{42} = a_{42} & \quad a_{42} = \frac{1}{[120 w_4 c_2 (c_4 - c_3)]} \cdot \left[ \frac{(c_3 - c_4)(31 - 60c_3)}{3(c_3 - c_2)} \right] \\
        & \quad + \frac{(31 - 40c_4)}{2} \\
    d_{43} = a_{43} & \quad a_{43} = \frac{31 - 60c_4}{360 w_4 c_2 (c_3 - c_2)} \\
    u_1 = 1 - w_1 & \quad w_1 = \frac{3}{2} - (w_2 + w_3 + w_4) \\
    u_2 = -w_2 & \quad w_2 = \frac{31 + 50c_3 c_4 - 40(c_3 + c_4)}{120c_2 (c_2 - c_3)(c_2 - c_4)} \\
    u_3 = -w_3 & \quad w_3 = \frac{31 + 50c_2 c_4 - 40(c_2 + c_4)}{120c_3 (c_3 - c_2)(c_3 - c_4)} \\
    u_4 = w_4 & \quad w_4 = \frac{31 + 50c_2 c_3 - 40(c_2 + c_3)}{120c_4 (c_4 - c_2)(c_4 - c_3)} \\
\end{align*}
\]

\(c_2\) and \(c_3\) free, \(c_2 = .25\) and \(c_3 = .869\) give an optimum method.

Again, eq. A.117 would be faster than comparable fifth order RKE's
but not quite as accurate.
There would be very little advantage in deriving an implicit or semi-implicit RKMS form ($M_i = V$ or $M_i = i$ in eqs. A.112 or A.113), because of the additional complexities involved.

A.5 Special Types or Hybrid Runge-Kutta Methods

Two most widely used methods of numerically solving ODE's are the Runge-Kutta methods and the predictor-corrector methods.

For complex systems of ODE's numerical solutions, using the usual Runge-Kutta's discussed so far, have one big disadvantage--function evaluations take up the major portion of the computational time involved. Multi-step predictor-corrector methods require considerably less function evaluations per step. So obviously reducing the number of function evaluations ($V$) of a Runge-Kutta, while retaining some or all the good points of a Runge-Kutta, would yield good methods. Some compromise between Runge-Kutta's and predictor-correctors is indicated.

Single step Runge-Kutta's have certain definite advantages and disadvantages, and so do the predictor-corrector multi-step method.

**Characteristics of Comparable Order Methods**

<table>
<thead>
<tr>
<th>Single Step Runge-Kutta</th>
<th>Multi-step predictor-correctors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advantages:</td>
<td>Disadvantages:</td>
</tr>
<tr>
<td>1) Self-starting</td>
<td>1) Not self-starting</td>
</tr>
<tr>
<td>2) Comparatively good stability characteristics because single step (so large $h$ possible)</td>
<td>2) Comparatively bad stability characteristics because multi-step ($h$ comparatively small)</td>
</tr>
<tr>
<td>3) Step size and order easily changeable at each step.</td>
<td>3) Step size and order change not easily instituted at each step.</td>
</tr>
</tbody>
</table>
Disadvantages:
1) May require greater computational time as comparatively large number of function evaluations required
2) No easily computable and accurate error estimate available
3) High order methods involve extremely complicated deriving equations.

Advantages:
1) Computationally fast as comparatively small number of function evaluations required (provided not too many iterations done).
2) Fairly easily computable and accurate error estimate usually available.
3) High order methods easily derived/

Hybrid Runge-Kutta's have been developed to combine the advantages of the Runge-Kutta and predictor-corrector methods, while trying to avoid their disadvantages.

Rosser 63 developed a class of ingenious hybrid Runge-Kutta's with great potential. These methods favorably combine the advantageous features of Runge-Kutta's and predictor-correctors. They may be described as implicit, multi-step, predictor-corrector Runge-Kutta's.

These methods are "Block" methods and proceed by blocks of N steps. Each block is completely independent of other blocks. The solutions from previously evaluated steps are only required from, and used within, a particular block. A large degree of flexibility is built into these block methods giving them three concrete advantages:
1) Step size is constant within a block but can easily be changed from block to block
2) Order of accuracy can be varied from block to block by changing
N, the number of steps constituting a particular block; and by changing the number of iterations used in each block.

3) Accurate estimates of errors incurred within a block are available and so give good pointers for the best values of h and N for the next block.

These "Block" methods result in a considerable savings of function evaluations and are hence highly recommended. The Rosser "Block" Runge-Kutta's can be defined using the following formulation:

For a block of N steps:

\[ x_1 - x_0 = x_2 - x_1 = \cdots = x_n - x_{n-1} = h > 0 \]

The subscript 0 denotes the starting or initial step, hence \( x_0, y_0 \) are known (from initial values or the results of the previous block).

\[
\begin{align*}
y_0 &= y_0 & \text{Starting values for a block} & \text{A.122} \\
y_0' &= f(x_0, y_0) \\
y_m' &= y_0' + mhf(x_0, y_0) & \text{A.123} \\
y^r_m &= f(x_m, y^r_m) & \text{A.124} \\
r \geq 0 & \quad m = 1, 2, 3, \cdots, N
\end{align*}
\]

The superscript \( r \) is the iterate count.

\[
\begin{align*}
y_1^{r+1} &= y_0 + h[y_0' + a_{11}y_1^r + \cdots + a_{1m}y_m^r + \cdots + a_{1n}y_n^r] \\
& \quad \vdots \\
y_m^{r+1} &= y_0 + h[y_m' + a_{m1}y_1^r + \cdots + a_{mm}y_m^r + \cdots + a_{mn}y_n^r] & \text{A.125} \\
& \quad \vdots \\
y_n^{r+1} &= y_0 + h[y_n' + a_{n1}y_1^r + \cdots + a_{nm}y_m^r + \cdots + a_{nn}y_n^r] \\
r \geq 0 & \quad m = 1, 2, 3, \cdots, N
\end{align*}
\]
$w_1, w_2 \ldots w_n$ and $a_{11}, a_{1m} \ldots a_{nn}$ are constants defining a particular block method.

The working of the block method is easily understood. Consider the $r=0$ case, i.e., the first iteration: $y_1^1$ is calculated from eq. A.125; this requires $N$ function evaluations—$y_1^0, y_2^0, \ldots y_n^0$ which are calculated from eqs. A.122 and A.124 ($y_0^1$ is assumed known from a previous block and remains the same for any iteration; though $y_0^0$ must be calculated for the first block from eq. A.122 and hence the first block needs $N+1$ function evaluations). Similarly $y_2^1$, $y_3^1 \ldots y_n^1$ are calculated from eq. A.125, these use the same $y_1^0, y_2^0, \ldots y_n^0$ used by $y_1^1$. Thus a complete sweep for a block (one complete iteration of eq. A.125) requires $N$ function evaluations. For $r=1$, i.e., the second iteration: $y_1^2$ is calculated from eq. A.125 in the same way as for the $r=0$ case: the values $y_1^1, y_2^1, \ldots y_n^1$ are substituted in eq. A.124 to yield $y_1^1, y_2^1, \ldots y_n^1$, which are then used in eq. A.125. $y_2^2, y_3^2, \ldots y_n^2$ are similarly calculated.

Up to a point increasing $r$, the number of iterations, increases the order of accuracy of the solutions $y_1^1, y_2^1 \ldots y_n^1$. For example, for a $N=4$ method (Milne method) with $0 \leq r \leq 4$ each iteration gives $y_m^r$ to an order of accuracy of $4r+2$. For $r > 4$ no improvement in order of accuracy is gotten. If $r=3$ is used a fifth order method results and uses a total of 16 ($4N$) function evaluation for a block of four steps ($N=4$). Thus a fifth order Runge-Kutta is gotten which uses four function evaluations per step. If $r=4$ is used a sixth order Runge-Kutta results which uses a total of twenty ($5N$) function evaluations for a block of four steps ($N=4$). Thus a sixth order Runge-Kutta is gotten which uses five function evaluations per step.
The above discussion illustrates the flexibility of these "Block" Runge-Kutta's—high order of accuracy for the number of function evaluations, and variable order of accuracy possible.

Rosser 63 has derived several such methods where the number of function evaluations are further reduced by curtailing some of the early steps, and accelerating the convergence of the later steps. In these methods order of accuracy is changeable, and is changed by varying N or r. An error estimate is also given, and a procedure for changing N and h from block to block outlined. In general if $N = 2r$ or $2r + 1$ a Runge-Kutta of order $2r + 1$ results.

The Block approach improves conventional Runge-Kutta's and makes them comparable to predictor-correctors on speed. But the "Block" approach is applicable to predictor-correctors also and will improve them too.

Rosser has derived methods up to $N \leq 8$, i.e., up to a tenth order Runge-Kutta. In the following formulae $y^{(s)}$ is used in the error estimates and is $\frac{d^s y}{dx^s}$ evaluated in $[x_0, x_N]$. 
One-point formulas:

\[ y_0 - y_o = h y_o' + \frac{h^3 y^{(4)}}{24} \text{.} \tag{A.126} \]

Two-point formulas:

\[ y_1 - y_o = \frac{h}{2} (y_o' + y_1') - \frac{h^3 y^{(4)}}{12} \text{,} \tag{A.127} \]

\[ y_2 - y_o = 2 hy_1' + \frac{h^3 y^{(4)}}{3} \text{.} \]

Three-point formulas:

\[ y_3 - y_o = \frac{h}{12} (5y_o' + 8y_1' - y_2') + \frac{h^3 y^{(4)}}{24} \text{,} \tag{A.128} \]

\[ y_4 - y_o = \frac{h}{3} (y_o' + 4y_1' + y_2') - \frac{h^3 y^{(4)}}{90} \text{,} \]

\[ y_5 - y_o = \frac{3h}{8} (y_o' + 3y_1' + 3y_2' + y_3') - \frac{3h^3 y^{(4)}}{80} \text{.} \]

Four-point formulas:

\[ y_6 - y_o = \frac{h}{24} (9y_o' + 19y_1' - 5y_2' + y_3') - \frac{19h^3 y^{(4)}}{720} \text{,} \tag{A.129} \]

\[ y_7 - y_o = \frac{h}{3} (y_o' + 4y_1' + y_2') - \frac{h^3 y^{(4)}}{90} \text{,} \]

\[ y_8 - y_o = \frac{3h}{8} (y_o' + 3y_1' + 3y_2' + y_3') - \frac{3h^3 y^{(4)}}{80} \text{,} \]

\[ y_9 - y_o = \frac{4h}{3} (2y_1' - y_2' + 2y_3') + \frac{14h^3 y^{(4)}}{45} \text{,} \]

\[ y_{10} - y_o = \frac{h}{32} (10y_o' - 81y_1' + 136y_2' + 31y_3') - \frac{351h^3 y^{(4)}}{160} \text{.} \]

Five-point formulas:

\[ y_{11} - y_o = \frac{h}{720} (251y_o' + 646y_1' - 264y_2' + 106y_3' - 19y_4') + \frac{3h^3 y^{(4)}}{160} \text{,} \tag{A.130} \]

\[ y_{12} - y_o = \frac{h}{90} (29y_o' + 124y_1' + 24y_2' + 4y_3' - y_4') + \frac{h^3 y^{(4)}}{90} \text{,} \]

\[ y_{13} - y_o = \frac{3h}{60} (9y_o' + 34y_1' + 24y_2' + 14y_3' - y_4') + \frac{3h^3 y^{(4)}}{160} \text{,} \]

\[ y_{14} - y_o = \frac{2h}{45} (7y_o' + 32y_1' + 12y_2' + 32y_3' + 7y_4') - \frac{8h^3 y^{(4)}}{915} \text{,} \]

\[ y_{15} - y_o = \frac{5h}{144} (19y_o' - 10y_1' + 120y_2' - 70y_3' + 85y_4') + \frac{95h^3 y^{(4)}}{288} \text{,} \]

\[ y_{16} - y_o = \frac{h}{540} (3329y_o' + 1530y_1' - 1216y_2' + 450y_3' - 31y_4') + \frac{16h^3 y^{(4)}}{45} \text{.} \]
\[ y_{n+1} = y_n + \frac{h}{540} \left( -31y_n' + 459y_n'' - 1216y_n''' + 1530y_n'''' + 329y_n'''''ight) \]
\[ - \frac{16h^5 y^{(4)}}{45}, \]
\[ y_{n+1} = y_n + \frac{h}{20} \left( 11y_n' + 81y_n'' - 64y_n''' + 81y_n'''' + 11y_n''''' \right) - \frac{9h^3 y^{(3)}}{35}. \]

Six-point formulas:
\[ y_{n+1} = y_n + \frac{h}{1440} \left( 475y_n' + 1427y_n'' - 798y_n''' + 482y_n'''' - 173y_n''''' + 27y_n'''''' \right) - \frac{863h^4 y^{(4)}}{60480}, \]
\[ y_{n+1} = y_n + \frac{h}{90} \left( 28y_n' + 129y_n'' + 14y_n''' + 14y_n'''' - 6y_n''' + y_n'''' \right) - \frac{37h^3 y^{(3)}}{3780}, \]
\[ y_{n+1} = y_n + \frac{3h}{160} \left( 17y_n' + 73y_n'' + 38y_n''' + 38y_n'''' - 7y_n''''' + y_n'''''' \right) - \frac{29h^2 y^{(2)}}{2240}, \]
\[ y_{n+1} = y_n + \frac{2h}{45} \left( 7y_n' + 32y_n'' + 12y_n''' + 32y_n'''' + 7y_n''''' \right) - \frac{8h y^{(1)}}{945}, \]
\[ y_{n+1} = y_n + \frac{5h}{288} \left( 19y_n' + 75y_n'' + 50y_n''' + 50y_n'''' + 75y_n''''' + 19y_n'''''' \right) - \frac{275h^2 y^{(2)}}{12096}, \]
\[ y_{n+1} = y_n + \frac{3h}{10} \left( 11y_n' - 14y_n'' + 26y_n''' - 14y_n'''' + 11y_n''''' \right) + \frac{41h y^{(1)}}{140}. \]

Seven-point formulas:
\[ y_{n+1} = y_n + \frac{h}{60480} \left( 19087y_n' + 65112y_n'' - 46461y_n''' - 37554y_n'''' + 20211y_n''''' + 6312y_n'''''' + 863y_n''''''' \right) - \frac{275h^3 y^{(3)}}{24192}, \]
\[ y_{n+1} = y_n + \frac{h}{3780} \left( 1139y_n' + 5640y_n'' + 33y_n''' + 1328y_n'''' - 807y_n''''' + 264y_n'''''' - 37y_n'''''''} + \frac{8h^2 y^{(2)}}{945}, \]
\[ A.131 \]
\[ A.132 \]
\[ y_1 - y_0 = \frac{h}{12096} (687y_0' + 3240y_1' + 1161y_2' + 2176y_3' \\
- 729y_4' + 216y_5' - 29y_6') + \frac{9h^4 y^{(4)}}{896}, \]
\[ y_4 - y_0 = \frac{2h}{945} (143y_0' + 696y_1' + 192y_2' + 752y_3' \\
+ 87y_4' + 24y_5' - 4y_6') + \frac{8h^4 y^{(4)}}{945}, \]
\[ y_5 - y_0 = \frac{5h}{12096} (743y_0' + 3480y_1' + 1275y_2' + 3200y_3' \\
+ 2325y_4' + 1125y_5' - 55y_6') + \frac{275h^6 y^{(6)}}{24192}, \]
\[ y_6 - y_0 = \frac{h}{140} (41y_0' + 216y_1' + 27y_2' + 272y_3' \\
+ 27y_4' + 216y_5' + 41y_6') - \frac{9h^4 y^{(4)}}{1400}, \]
\[ y_7 - y_0 = \frac{7h}{3654} (751y_0' - 840y_1' + 8547y_2' - 11648y_3' \\
+ 14637y_4' - 7224y_5' + 4417y_6') + \frac{5257h^6 y^{(6)}}{17280}, \]
\[ y_8 - y_0 = \frac{4h}{945} (115y_0' + 1312y_1' - 2048y_2' + 3132y_3' \\
- 2048y_4' + 1312y_5' + 115y_6') - \frac{294h^4 y^{(4)}}{14175}. \]

Eight-point formulas:
\[ y_1 - y_0 = \frac{h}{12096} (36799y_0' + 139849y_1' - 121797y_2' + 123133y_3' \\
- 88547y_4' + 41499y_5' - 11351y_6' + 1875y_7') - \frac{33933h^6 y^{(6)}}{3628800}, \]
\[ y_2 - y_0 = \frac{h}{9750} (1107y_0' + 5864y_1' - 639y_2' + 2448y_3' \\
- 1927y_4' + 936y_5' - 261y_6' + 32y_7') - \frac{119h^4 y^{(4)}}{16200}, \]
\[ y_3 - y_0 = \frac{h}{4480} (1325y_0' + 6795y_1' + 1377y_2' + 5927y_3' \\
- 3033y_4' + 1377y_5' - 373y_6' + 43y_7') - \frac{360h^6 y^{(6)}}{44800}, \]
\[ y_4 - y_0 = \frac{2h}{945} (139y_0' + 724y_1' + 108y_2' + 802y_3' \\
- 53y_4' + 108y_5' - 32y_6' + 4y_7') - \frac{107h^4 y^{(4)}}{14175}, \]

A. 133
\[ y_5 - y_0 = \frac{5h}{145152} \left( 8341y_0' + 46030y_1' + 1510y_2' + 63670y_3' \\
- 800y_4' + 34186y_5' - 9830y_6' + 2290y_7' - 245y_8' \right) \\
+ \frac{25h^{10}y^{(10)}}{3584}, \]

\[ y_6 - y_0 = \frac{h}{1400} \left( 401y_0' + 2232y_1' + 18y_2' + 3224y_3' \\
- 360y_4' + 2664y_5' + 158y_6' + 72y_7' - 9y_8' \right) \\
+ \frac{9h^{10}y^{(10)}}{1400}, \]

\[ y_7 - y_0 = \frac{7h}{518400} \left( 21361y_0' + 116662y_1' + 6958y_2' + 155134y_3' \\
+ 7840y_4' + 105154y_5' + 74578y_6' + 31882y_7' - 1169y_8' \right) \\
+ \frac{8183h^{10}y^{(10)}}{1036800}, \]

\[ y_8 - y_0 = \frac{4h}{14175} \left( 989y_0' + 5888y_1' - 928y_2' + 10496y_3' \\
- 4540y_4' + 10496y_5' - 928y_6' + 5888y_7' + 989y_8' \right) \\
- \frac{2368h^{11}y^{(11)}}{467775}. \]
\[
y_s - y_o = \frac{5h}{24192} (1431y_s' + 7345y_t' + 1395y_s' + 8325y_s') \\
+ 2725y_s' + 3411y_s' - 403y_s' + 535y_s' - \frac{1755y_s^{(9)}}{20736},
\]
\[
y_s - y_o = \frac{h}{140} (41y_o' + 216y_s' + 27y_s' + 272y_s') \\
+ 27y_s' + 216y_s' + 41y_s' - \frac{9h^9y_s^{(9)}}{1400},
\]
\[
y_t - y_o = \frac{7h}{17280} (751y_s' + 3577y_t' + 1323y_s' + 2989y_s') \\
+ 2989y_s' + 1323y_s' + 3577y_s' + 751y_t') - \frac{8183h^9y_t^{(9)}}{518400},
\]
\[
y_e - y_o = \frac{8h}{945} (460y_s' - 954y_s' + 2196y_s' - 2459y_s') \\
+ 2196y_s' - 954y_s' + 460y_s' + \frac{3956h^9y_s^{(9)}}{14175}.
\]

Nine-point formulas:
\[
y_1 - y_o = \frac{h}{3628800} (1070017y_o' + 4467094y_s' - 4604594y_t') \\
+ 5595358y_s' - 5033120y_s' + 3146338y_s' - 12912144y_s' \\
+ 312574y_t' - 33953y_t') + \frac{8153a^{19}y_t^{(19)}}{1036800},
\]
\[
y_2 - y_o = \frac{h}{113400} (32377y_o' + 182584y_s' - 42494y_t') \\
+ 120088y_s' - 116120y_s' + 74728y_s' - 31154y_s' \\
+ 7624y_t' - 833y_t') + \frac{9h^{19}y_s^{(19)}}{1400},
\]
\[
y_3 - y_o = \frac{h}{44800} (12881y_o' + 70002y_s' + 3438y_t' + 79934y_t') \\
- 56160y_s' + 34434y_s' - 14062y_s' + 3402y_t' - 360y_t') \\
+ \frac{25h^{19}y_s^{(19)}}{3384},
\]
\[
y_4 - y_o = \frac{h}{14175} (4063y_o' + 22576y_s' + 244y_s' + 32752y_t') \\
- 9080y_s' + 9232y_s' - 3056y_t' + 976y_t' - 107y_t') \\
+ \frac{94h^{19}y_s^{(19)}}{14175}.
\]
Butcher [56, 57] also creates hybrid methods which are a combination of RKMS's and predictor-corrector methods. Butcher [57] lists RKMS's with \( S_n = 2, 3, \) and \( 4 \) (see section A.4) and of order \( p = 2S_n + 2 \), with \( V = 4 \). Methods up to \( S_n = 15 \) are proved to exist. On basis of various tests these methods are highly recommended (they compare favorably with standard predictor-correctors). In the opinion of the author, the Rosser "Block" methods are, on the whole, better than these.

Gear [64], Gragg and Stetter [65], and Dahlquist [66] present methods similar to Butcher's methods.

Another special class of Runge-Kuttas is given in Gates [26]. Gates formulates explicit Runge-Kuttas in which each stage (each \( k_i \)) is independent of the other stages. In the conventional Runge-Kuttas each stage is not independent of the other stages. Each independent stage of the Gates formulae corresponds to a standard quadrative formula, e.g., Gauss, Radau, Lobatto, etc. Making each stage independent of the others has a penalty—\( V \) increases for a given order \( p \) as compared to the usual Runge-Kuttas. For example, for \( p = 4, 5 \), and \( 6 \) Gates forms have \( V = 5, 7 \) and \( 11 \) respectively. The advantage of Gates' formulation is the flexibility gotten by having the coefficients of one stage independent of another; another advantage is that deriving equations for these methods are much simpler than the usual RKES's and so high order methods can be derived fairly easily.

Stoller and Morrison [67] and Day [68] discuss similar quadrative Runge-Kuttas.

From the computational time point of view these methods are obviously not recommended.
General Comments for Further Development of the Runge-Kutta Class of Methods:

The previous sections have tried to sample various types of Runge-Kuttas available in current literature.

The field of Runge-Kuttas is wide open and improvements are possible. Runge-Kuttas in general arrive at an accurate solution at the end of an integration step, by combining solutions (k_i's) at intermediate steps which are not, in themselves so accurate. So one way of improving Runge-Kuttas would be to choose the coefficients of each stage in such a way that the k_i's would be accurate estimates of f(T,Y).

This should be possible without significantly increasing V, unlike the Gates [26] approach (see section A.5). Then as each k_i would be an accurate estimate of f(T,Y) a hermitian curve fit would be possible giving a higher order of accuracy.

The most obvious improvement needed by Runge-Kuttas is to reduce their stages V per step, this would make them comparable to predictor-correctors on computational time. In all other aspects (except error estimation), i.e., stability and self-starting features Runge-Kuttas are superior to predictor-correctors. So the second area of improvement lies in reducing V. This is possible to some extent by using the "Back Step" and "Front Step" methods outlined in section A.1-2, or by using the Shanks' technique outlined in section A.1-5. Conventional RKE's always have c_i=0 and usually have a c_i=1, hence the k_i corresponding to the c_i=1 of the (T_n,Y_n) step could be used for the k_i of the (T_{n+1},Y_{n+1}) step reducing V by one. This would cost something--accuracy, because the solution Y_n, available at the end of
\((T_n, Y_n)\) gives a more accurate estimate for \(k_i\) than is possible by using the \(k_i\) of the previous step. Also if two or more \(c_i\)'s have the same value, then only one \(k_i\) corresponding to the first of these \(c_i\)'s need be evaluated. For the rest of these \(k_i\)'s no more function evaluations are necessary, they can use the same value as the first of these \(k_i\)'s. Again \(V\) is reduced but a loss in accuracy would occur. For example, a RKE(4,4) which has \(c_1=0, c_2=\frac{1}{2}, c_3=\frac{1}{2}, c_4=1\) would become a RKE(4,2) by using the two techniques outlined above—\(k_i\) at \((T_{n+1}, Y_{n+1})\) would be the \(k_4\) of \((T_n, Y_n)\), \(k_3\) at \((T_{n+1}, Y_{n+1})\) would be \(k_2\) of the same step. It is not possible to say, off hand, whether the "Back Step" or "Front Step" technique would be more accurate than this one of interchanging \(k_i\)'s. Obviously the interchange techniques could also be used to further reduce the \(V\) of a "Back Step" or "Front Step" method, with (probably) a further loss of accuracy, or the Rosser [63] (see section A.5) "Block" technique could be used to further speed up these methods.

A third area of development (not speed improvement) lies in the RKS1 class of methods. Here a calculation procedure for \(f\) or a Jacobian is required. As this is available it might as well be used to improve the order of accuracy, though at the cost of extra computational effort.

\[
Y_{n+1} = Y_n + \sum_{i=1}^{V} w_i k_i + \sum_{i=1}^{V} u_i g_i \\
k_i = hf(T_n + c_i h, Y_n + \sum_{j=1}^{i} a_{ij} k_j) \quad A.135 \\
g_i = h^2 fy(T_n + d_i h, Y_n + \sum_{j=1}^{L_i} b_{ij} g_j) \\
L_i \leq V
\]
The Rosser [63] (see section A.5) should be applicable to speed up any type of Runge-Kutta.

Many authors have created hybrid methods which combine Runge-Kutta's with predictor-correctors, to try and combine the advantages of both methods (see section A.5). Similarly hybrid methods combining Runge-Kutta with extrapolation and other methods should be possible.

The Rosser [63] "Block" technique (see section A.5) could be adapted to further speed up these hybrid methods. These hybrid methods may grow to such forms that Runge or Kutta would hardly recognize their method.

There are other areas in which Runge-Kutta's can be developed and improved, it is hoped that present and future mathematicians will do so.
Bibliography


104. Radau, M., "Etude sur les Formules d'Approximation qui Servent a
Pures Appl., 6, 283 (1880).

105. Ralston, A. and Wilf, H., Mathematical Methods for Digital Com-


107. Richter, W., "Estimation de l'erreur Commise Dans la Methode
de M. W. E. Milne pour l'Integration d'un Systeme de n Equations

(1962).


111. Sammet, J., "Formula Manipulation Compiler," Datamation, 12,
32 (1966).

112. Sarafyan, D., "Determination of Optimal Step-Size for Runge-
Kutta Processes," Louisiana State Univ., Tech. Rep. No. 54,
Forthcoming.

113. Sarafyan, D., "Estimation of Errors for the Approximate Solution
of Differential Equations and Their Systems," Louisiana State Univ.,

114. Sarafyan, D., "Composite and Multistep Runge-Kutta Formulas,"

115. Scheid, F., Theory and Problems of Numerical Analysis, McGraw-


Equations not Containing the First Derivative Explicitly,"
Comp. J., 6, 368 (1963).

118. Sinha, K., "Laminar Incompressible Boundary Layer in a Divergent
Channel with Homogeneous Suction at the Wall," Nat. Inst. of
Sci. of India, Proceedings, Part A--Physical Sciences, Supplement


APPENDIX II
MULTI-STEP METHODS

This appendix lists many of the multi-step methods that are in the current literature.

The most common form of multi-step methods is

$$\sum_{i=0}^{k} (\alpha_i y_{n+1-i} + h \beta_i y'_{n+1-i}) = 0.$$  

When $\beta_0 = 0$, then the equation becomes an explicit equation for $y_{n+1}$; such equations are called Predictor (P) equations. When $\beta_0 \neq 0$, the equation becomes an implicit equation for $y_{n+1}$, i.e., the equation must be iterated for $y_{n+1}$. Such equations are called Corrector (C) equations.

Although the Predictor (P) equation can be used with one derivative evaluation (E) at each step of the numerical solution, usually a combination of the P and C equations is used to solve the initial value problem. The various modes of these P-C combinations are PECE, PEC, PE(CE)$^S$ and P(EC)$^S$. The first two modes are not iterative methods. First, $y_{n+1}$ is predicted. Then, there is a derivative evaluation of $y'_{n+1}$. Then $y_{n+1}$ is corrected using $y'_{n+1}$. For the first mode a new $y'_{n+1}$ is evaluated. The third and fourth modes simply iterate $s$ times on the corrector.

The Predictor and Corrector equations presented below are arranged in related groups. The format of each equation is in the following order: 1) the equation number $S$, 2) the order $m$ and number of backsteps $n$, written $(m,n)$, and 3) the $\alpha$ and $\beta$ coefficients. The general format is
\[ S. \ (m,n) \ \left( \frac{1}{a_0} \right) (a_1 + a_2 + \cdots + a_n) \]
\[ + \left( \frac{1}{\beta_0} \right) (\beta_0 + \beta_1 + \cdots + \beta_n) . \]

**Euler Predictor:** [556]

1. \( (1,1) \ \left( \frac{1}{1} \right) (1) + \left( \frac{1}{1} \right) (0 + 1) \)

**Milne Predictor:** [556]

2. \( (4,3) \ \left( \frac{1}{4} \right) (0 + 0 + 0 + 1) + \frac{1}{3} (0 + 8 - 4 + 8) \)

**Millman-Klopfenstein Predictor:** [277]

3. \( (4,4) \ \left( \frac{1}{4} \right) (-0.29 - 15.39 + 12.13 + 4.55) + \left( \frac{1}{4} \right) (0 + 2.27 + 6.65 + 13.91 + 0.69) \)

**Craine-Klopfenstein Predictor:** [95]

4. \( (4,4) \ \left( \frac{1}{4} \right) (1.547 - 1.867 + 2.017 - 0.6973) + \left( \frac{1}{4} \right) (0 + 2.002 - 2.031 + 1.818 - 0.7143) \)

**Hermite Extrapolation Predictors:** [556]

5. \( (3,2) \ \left( \frac{1}{3} \right) (-4 + 5) + \left( \frac{1}{3} \right) (0 + 4 + 2) \)

6. \( (4,3) \ \left( \frac{1}{4} \right) (-9 + 9 + 1) + \left( \frac{1}{4} \right) (0 + 6 + 6) \)

7. \( (5,3) \ \left( \frac{1}{5} \right) (-18 + 9 + 10) + \left( \frac{1}{5} \right) (0 + 9 + 18 + 3) \)

8. \( (7,4) \ \left( \frac{1}{7} \right) (-128 - 108 + 198 + 47) + \left( \frac{1}{7} \right) (0 + 16 + 72 + 48 + 4) \)

9. \( (9,5) \ \left( \frac{1}{9} \right) (-475 - 1400 + 600 + 1150 + 131) + \left( \frac{1}{9} \right) (0 + 25 + 200 + 300 + 100 + 5) \)
Nystrom Predictors: [556]

10. (2,1) \( \frac{1}{1}(0 + 1) + \frac{1}{1}(0 + 2) \)
11. (3,2) \( \frac{1}{1}(0 + 1) + \frac{1}{1}(0 + 2 + 0) \)
12. (4,3) \( \frac{1}{1}(0 + 1) + \frac{1}{3}(0 + 7 - 2 + 1) \)
13. (5,4) \( \frac{1}{1}(0 + 1) + \frac{1}{3}(0 + 8 - 5 + 4 - 1) \)
14. (6,5) \( \frac{1}{1}(0 + 1) + \frac{1}{90}(0 + 269 - 266 + 294 - 146 + 29) \)

Same Predictors from method of undetermined coefficient: [556]

15. (5,4) \( \frac{1}{1}(0 + 0 + 1) + \frac{1}{8}(0 + 21 - 9 + 15 - 3) \)
16. (5,4) \( \frac{1}{1}(0 + 0 + 0 + 1) + \frac{1}{24}(0 + 9 - 32 + 64) \)
17. (5,3) \( \frac{1}{1}(-9 + 9 + 1) + \frac{1}{1}(0 + 6 + 6) \)
18. (5,3) \( \frac{1}{1}(-8 + 9) + \frac{1}{3}(0 + 17 + 14 - 1) \)
19. (5,3) \( \frac{1}{1}(-7 + 9 - 1) + \frac{1}{3}(0 + 16 + 10 - 2) \)
20. (4,3) \( \frac{1}{1}(-54 + 45 + 10) + \frac{1}{1}(0 + 24 + 42) \)
21. (4,2) \( \frac{1}{1}(-4 + 5) + \frac{1}{1}(0 + 4 + 2) \)
22. (5,4) \( \frac{1}{1}(-1 + 0 + 1 + 1) + \frac{1}{3}(0 + 3 + 0 + 3) \)

Schoen Predictors: [461]

23. (5,5) \( \frac{1}{1}(-0.01745885 + 1.29864818 - 0.13318934 - .14799999) \)

\[ + \left( \frac{1}{1} \right)(0 + 3.0001344 - 3.05980708 + 2.98844518 - 1.66187410 \]
\[ + 0.32137111) \]
24. \((6,6)\) \(\left(\frac{1}{1}\right)(-.92790378 + 2.05094161 + .54664912 - 0.67100875 + 0.0005296 + 0.00126884 + \left(\frac{1}{1}\right)(0 + 3.60328436 - 3.60284254 + 5.40345116 - 4.77344249 + 1.80481406 - 0.29749495)\)

Krough Predictors: [556]

25. \((4,4)(A)\) \(\left(\frac{1}{2}\right)(1 + 1) + \left(\frac{1}{48}\right)(0 + 119 - 99 + 69 - 17)\)

26. \((4,4)(B)\) \(\left(\frac{1}{7}\right)(4 + 3) + \left(\frac{1}{42}\right)(0 + 103 - 88 + 61 - 15)\)

27. \((5,5)\) \(\left(\frac{1}{31}\right)(-1 + 32) + \left(\frac{1}{7440}\right)(0 + 22,321 - 21,774 + 24,216 - 12,034 + 2391)\)

28. \((6,6)\) \(\left(\frac{1}{12}\right)(-11 + 23) + \left(\frac{1}{17,280}\right)(0 + 62,248 - 62,255 + 101,430 - 76,490 + 30,545 - 5079)\)

29. \((7,7)\) \(\left(\frac{1}{10}\right)(-21 + 31) + \left(\frac{1}{604,800}\right)(0 + 2,578,907 - 2,454,408 + 5,615,199 - 5,719,936 + 3,444,849 - 1,149,048 + 164,117)\)

Adams-Bashforth Predictors: [556]

30. \((2,2)\) \(\left(\frac{1}{1}\right)(1) + \left(\frac{1}{2}\right)(0 + 3 - 1)\)

31. \((3,3)\) \(\left(\frac{1}{1}\right)(1) + \left(\frac{1}{12}\right)(0 + 23 - 16 + 5)\)

32. \((4,4)\) \(\left(\frac{1}{1}\right)(1) + \left(\frac{1}{24}\right)(0 + 55 - 59 + 37 - 9)\)

33. \((5,5)\) \(\left(\frac{1}{1}\right)(1) + \left(\frac{3}{720}\right)(0 + 1901 - 2774 + 2616 - 1274 + 251)\)

34. \((6,6)\) \(\left(\frac{1}{1}\right)(1) + \left(\frac{1}{1440}\right)(0 + 4277 - 7923 + 9982 - 7298 + 2887 - 475)\)
35. \((7,7)\) \(\frac{1}{4}(1) + \left(\frac{1}{60,480}\right)(0 + 198,721 - 447,288 + 705,549 - 688,256 + 407,139 - 134,472 + 19,087)\)

36. \((8,8)\) \(\frac{1}{4}(1) + \left(\frac{1}{120,960}\right)(0 + 434,241 - 1,162,169 + 2,183,877 - 2,664,477 + 2,102,243 - 1,041,723 + 295,767 - 36,799)\)

Milne Corrector: [556]

37. \((4,2)\) \(\left(\frac{1}{4}\right)(0 + 1) + \left(\frac{1}{3}\right)(1 + 4 + 1)\)

Hamming Corrector: [556]

38. \((4,2)\) \(\left(\frac{1}{8}\right)(9 + 0 - 1) + \left(\frac{3}{8}\right)(3 + 6 - 3)\)

Milne-Reynolds Corrector: [361]

39. \((5,3)\) \(\left(\frac{1}{8}\right)(1 + 7) + \left(\frac{1}{192}\right)(65 + 243 + 51 + 1)\)

Correctors from method of undetermined coefficients: [556]

40. \((4,2)\) \(\left(\frac{1}{3}\right)(4 + 1) + \left(\frac{1}{3}\right)(2 + 4)\)

41. \((5,3)\) \(\left(\frac{1}{1}\right)(0 + 0 + 1) + \left(\frac{1}{8}\right)(3 + 9 + 9 + 3)\)

42. \((5,3)\) \(\left(\frac{1}{3}\right)(1 + 1 + 1) + \left(\frac{1}{36}\right)(13 + 39 + 15 + 5)\)

43. \((5,3)\) \(\left(\frac{1}{2}\right)(1 + 1) + \left(\frac{1}{48}\right)(17 + 51 + 3 + 1)\)

44. \((5,3)\) \(\left(\frac{1}{3}\right)(2 + 1) + \left(\frac{1}{72}\right)(25 + 91 + 43 + 9)\)

45. \((5,3)\) \(\left(\frac{1}{17}\right)(9 + 9 - 1) + \left(\frac{1}{17}\right)(6 + 18)\)

46. \((5,3)\) \(\left(\frac{1}{9}\right)(9 + 1 - 1) + \left(\frac{1}{27}\right)(10 + 22 - 8)\)
47. 
\((5,3)\) \(\frac{1}{7}(9 - 1 - 1) + \frac{1}{21}(8 + 14 - 10)\)

48. 
\((5,3)\) \(\frac{1}{31}(45 - 9 - 5) + \frac{1}{31}(12 + 18 + 18)\)

49. 
\((5,3)\) \(\frac{1}{5}(9 - 3 - 1) + \frac{1}{5}(2 + 2 - 4)\)

Norsett A\((\alpha)\) stable Correctors: \([389]\]

50. 
\((4,3)\) \(\frac{1}{11}(18 - 9 + 2) + \frac{1}{11}(6)\)

51. 
\((5,4)\) \(\frac{1}{25}(48 - 36 + 16 - 3) + \frac{1}{25}(12)\)

52. 
\((6,5)\) \(\frac{1}{137}(300 - 300 + 200 - 75 + 12) + \frac{1}{137}(60)\)

53. 
\((7,6)\) \(\frac{1}{147}(360 - 450 + 400 - 225 + 72 - 10) + \frac{1}{147}(60)\)

Adams-Moulton Correctors: \([422]\]

54. 
\((2,1)\) \(\frac{1}{1}(1) + \frac{1}{2}(1 + 1)\)

55. 
\((3,2)\) \(\frac{1}{12}(1) + \frac{1}{12}(5 + 8 - 1)\)

56. 
\((4,3)\) \(\frac{1}{1}(1) + \frac{1}{24}(9 + 19 - 5 + 1)\)

57. 
\((5,4)\) \(\frac{1}{1}(1) + \frac{1}{720}(251 + 646 - 264 + 106 - 19)\)

58. 
\((6,5)\) \(\frac{1}{1}(1) + \frac{1}{1440}(475 + 1427 - 798 + 482 - 173 + 27)\)

59. 
\((7,6)\) \(\frac{1}{1}(1) + \frac{1}{60,480}(19,087 + 65,912 - 46,461 + 37,504 - 20,211 + 6312 - 863)\)

60. 
\((8,7)\) \(\frac{1}{1}(1) + \frac{1}{120,960}(36,799 + 139,849 - 121,797 + 123,133 - 88,547 + 41,499 - 11,351 + 1375)\)
61. \((9, 8)\) \(\left(\frac{1}{1}\right)(1) + \left(\frac{1}{3,628,800}\right)(1,070,017 + 4,467,094 - 4,604,594 + 5,593,358 - 5,033,120 + 3,146,338 - 1,291,214 + 312,874 - 339,533)\)

62. \((10, 9)\) \(\left(\frac{1}{1}\right)(1) + \left(\frac{1}{7,257,600}\right)(2,082,753 + 2,449,717 - 11,271,304 + 16,002,320 - 17,283,646 + 13,510,082 - 7,394,032 + 2,687,864 - 583,435 + 57,281)\)

Fehlberg Correctors: [422]

63. \((3, 2)\) \(\left(\frac{1}{5}\right)(4 + 1) + \left(\frac{1}{3}\right)(2 + 4)\)

64. \((4, 3)\) \(\left(\frac{1}{17}\right)(9 + 9 - 1) + \left(\frac{1}{17}\right)(6 + 18)\)

65. \((5, 4)\) \(\left(\frac{1}{27}\right)(16 + 0 + 0 + 11) + \left(\frac{1}{3}\right)(3 + 10 + 0 + 6 + 1)\)

66. \((6, 5)\) \(\left(\frac{1}{21,319}\right)(0 + 0 + 13,300 + 8775 - 756) + \left(\frac{1}{21,319}\right)(6720 + 29,700 + 13,500 + 21,300)\)

67. \((7, 6)\) \(\left(\frac{1}{1000}\right)(243 + 0 + 125 + 0 + 0 + 632) + \left(\frac{1}{400}\right)(120 + 567 + 0 + 600 + 0 + 405 + 72)\)

68. \((8, 7)\) \(\left(\frac{1}{82,490,048}\right)(0 + 0 + 28,107,625 + 0 + 29,545,048 + 2,783,200 - 1,994,625) + \left(\frac{1}{82,490,048}\right)(24,299,520 + 126,015,750 + 21,168,000 + 0 + 70,634,970)\)

Wesson Correctors: [422]

69. \((3, 2)\) \(\left(\frac{1}{2}\right)(1 + 1) + \left(\frac{1}{8}\right)(3 + 8 + 1)\)

70. \((3, 2)\) \(\left(\frac{1}{25}\right)(9 + 16) + \left(\frac{1}{300}\right)(109 + 328 + 55)\)
71. (5, 4) \(\left(\frac{1}{16}\right)(1 + 0 + 0 + 15) + \left(\frac{1}{11,520}\right)(3611 + 16,006 + 5496 + 15,466 + 3341)\)

72. (5, 4) \(\left(\frac{1}{16}\right)(8 + 2 + 1 + 5) + \left(\frac{1}{2304}\right)(767 + 2638 + 168 + 1282 + 185)\)

73. (5, 5) \(\left(\frac{1}{1}\right)(1) + \left(\frac{1}{2,440,080}\right)(859,838 + 2,143,299 - 802,706 + 267,244 - 18,396 - 9199)\)

74. (8, 8) \(\left(\frac{1}{1}\right)(1) + \left(\frac{1}{3,628,800}\right)(1,147,591 + 3,846,502 - 2,432,522 + 1,251,214 + 397,060 - 1,197,806 + 880,858 - 307,718 + 43,621)\)

Rodabaugh-Wesson Correctors: [422]

75. (5, 4) \(\left(\frac{1}{16}\right)(1 + 2 + 4 + 9) + \left(\frac{1}{11,520}\right)(3703 + 15,518 + 6168 + 10,898 + 1873)\)

76. (7, 6) \(\left(\frac{1}{64}\right)(1 + 2 + 4 + 8 + 16 + 33) + \left(\frac{1}{430,080}\right)(128,627 + 642,168 + 130,167 + 693,632 + 143,137 + 399,240 + 61,469)\)

77. (8, 8) \(\left(\frac{1}{1}\right)(1) + \left(\frac{1}{3,628,800}\right)(1,111,267 + 413,709 - 3,449,594 + 3,285,358 - 2,145,620 + 836,338 - 136,214 - 17,126 + 7297)\)

78. (9, 8) \(\left(\frac{1}{2,560,016}\right)(9784 + 20,133 + 41,040 + 79,775 + 159,816 + 319,691 + 639,792 + 1,289,985 + \left(\frac{1}{2,560,016}\right)(725,340 + 4,150,740 - 280,710 + 6,541,620 - 1,808,250 + 5,630,940 + 244,290 + 2,458,620 + 345,330)\)